

UNISYS**Firmware Download****Format Specification****3486 8968 Revision B****April 12, 1996**

Preliminary
Specification
Layout

for Upgrade and new
program for DFAST

APRIL 1996

(E 1)

FIRMWARE DOWNLOAD FORMAT SPECIFICATION

DOCUMENT CONTROL INFORMATION

CEDB controls the latest revision of this document. Consult Engineering Records of the DCL for assistance.

Engineering signatures appear in the EIR releasing this document.

This document, including the information contained herein, is restricted, confidential and proprietary to Unisys Corporation, and is to be used only by and disclosed only to those within Unisys Corporation with a need-to-know. Release outside Unisys Corporation requires authorization in writing by a level C or higher Executive of Unisys Corporation or by a designee thereof.

FIRMWARE DOWNLOAD FORMAT SPECIFICATION**CONTENTS**

1. SCOPE	5
2. PURPOSE	6
3. PROCEDURE	7
4. FORMAT	8
4.1 File Header Record	8
4.2 Field Definitions- Default	9
4.3 Field Definitions - Non-Mode 5 Based	11
4.4 Firmware Binary Data	12
5. APPENDIX	13
5.1 Method of Downloading Firmware	13
5.2 Default Download	13
5.3 Mode and Offset Controlled Download	13
5.3 Determining the Method of Download	15

FIRMWARE DOWNLOAD FORMAT SPECIFICATION

TABLES

Table 5-1. Download with Offset, CDB #1	14
Table 5-2. Download with Offset, CDB #2	14
Table 5-3. Download with Offset, CDB #3	15
Table 5-4. Download with Offset, CDB #4	15
Table 5-5. Download with Offset and Save, Last CDB.....	15

FIRMWARE DOWNLOAD FORMAT SPECIFICATION**1. SCOPE**

This document outlines in detail the format specification of download firmware file format used by Unisys to download firmware.

Revision B enables the ability to determine precisely what type of download command sequence a device may require. The modifications for Revision B do not require a new utility for existing download files unless the utility checks for non-blank data after the last Drive ID entry.

This specification is not meant to define vendor unique tape loading or other non-Write Buffer command based loading methods

FIRMWARE DOWNLOAD FORMAT SPECIFICATION

2. PURPOSE

The purpose of this document is to standardize the format of the firmware file which will simplify the task of reading the firmware file by the system application programs.

In addition, by standardizing the format of the download firmware file, other future peripheral vendors with download features could be supported by Unisys with minimum modifications required to the system application program. This document supersedes internal document R210.

FIRMWARE DOWNLOAD FORMAT SPECIFICATION**3. PROCEDURE****SCSI Firmware Download File Format**

The firmware data will be provided on a DOS 3.5 " diskette (1.44MB) in binary format. Physically, the media will contain 8K (1K=1024) byte contiguous records. The file shall have two logical record types: a header record which will contain all the information needed to interpret the content and perform some checks on the drives, and a body record which will contain all the firmware code. Physically, the first 8K record on the media will contain the header record. All other 8K records on the media will contain the firmware information. If the firmware size is not a multiple of 8K, then the last 8K record will contain extra storage (00 hex).

FIRMWARE DOWNLOAD FORMAT SPECIFICATION

4. FORMAT

4.1 File Header Record

The 8,192 byte firmware file header record format will be as follows:

Field Name	Size (in bytes)	Offset
Customer ID	8	0
Size of header record	8	8
Size of firmware data	8	16
New firmware level	4	24
Vendor ID	8	28
Number of drive ID	4	36
Number of replaceable firmware levels	4	40
Firmware level #1 record (most recent)	8	44
Firmware level #2 record	8	52
... (until the last level)	8	...
Drive ID #1	16	... +8
Drive ID #2	16	... + 16
... (until the last Drive ID #n)	16	... +... + 32
MODE	4	... +... + 48
Mode-Start	1	... +... + 52
Mode-Last	1	... +... + 53
Mode Offset Minimum Increment	8	... +... + 54
Mode Offset Maximum Increment	8	... +... + 62

Example Header

```

0:    UNISYS 00008192
16:   01048828 2 bFUJI
32:   TSU 00010001*
48:   2 AM2488D M248
64:   8  MODE67000005
80:   1201048848

```

*Note: The * in the above Header example is supposed to be the value 00 in hex, but there is no way in the ASCII command set to denote that character for the purposes of the example. Also, the above example is for the non-default case of a non-Mode 5 download sequence.*

FIRMWARE DOWNLOAD FORMAT SPECIFICATION**Unused Bytes in the 8K Header**

Typically, there will be a significant amount of unused data at the end of the Header Record. This unused data should be set to 00 hex. Further it is recommended that at least 16 bytes after the last valid entry of header data be set to 00 hex to allow for a break point in the flow of header information.

4.2 Field Definitions- Default

The default case for the download sequence uses a single Write Buffer command with Mode 5.

All letter representation is in upper case.

All data is in ASCII format unless otherwise indicated.

Customer ID

This field is always set to "UNISYS" and is left justified. Unused portions are ASCII blanks 20h.

EXAMPLE: "UNISYS "

Size of Header Record

This field is always set to 8K bytes and is right justified with leading zeros.

EXAMPLE: "00008192"

Size of Firmware Data

This field contains the number of decimal bytes (represented in ASCII) of firmware data and is right justified with leading zeros.

EXAMPLE: "00305152"

This value would convert to 4A800 hex bytes for use in the CDB.

New Firmware Level

Level of firmware as described in bytes 32 through 35 of SCSI INQUIRY data.

EXAMPLE: "0030"

Vendor ID

The name of the drive manufacturer, typically UNISYS for disk products, which corresponds to data returned from bytes 8 through 15 of drive SCSI INQUIRY command. This is specified in the procurement specification for that device and is left justified. Unused portions are ASCII blanks 20h.

EXAMPLE: "UNISYS "

FIRMWARE DOWNLOAD FORMAT SPECIFICATION**Number of Drive ID**

Number of different drive model numbers that use the same firmware and is right justified with leading zeros.

EXAMPLE: "0002"

Number of Replaceable Firmware Levels

Number of all prior firmware levels that were approved for production. This field is right justified with leading zeros.

EXAMPLE: "0002"

NOTE TO IMPLEMENTORS OF DOWNLOADING UTILITY:

If you are downloading a current version on a drive already containing that version of firmware, the use of the "New firmware Level" field data is recommended for verifying the correct version of firmware that is downloaded to the drive.

Firmware Level #X Record Format

8 bytes: numbered 7..0 with left most being byte 7.

byte 7: a format flag meaning,

if = "Hex 00"

"No format needed after the download of this level of firmware", or not a disk device

else

"Format needed after the download" (disk devices only)

byte 6..4:

Reserved (ASCII blanks 20h)

byte 3..0:

firmware level, as it appears in bytes 32 through 35 of the SCSI Inquiry data (Right Justified) for firmware level #X.

Note: Because the firmware level in this field is used to determine whether or not new firmware needs to be loaded, each new release of drive firmware must be uniquely identifiable in bytes 32 through 35 of the SCSI Inquiry data.

Drive ID # Record

Represents the drive model number or "PRODUCT IDENTIFICATION FIELD" as returned from bytes 16 through 31 of SCSI INQUIRY command and is left justified (ASCII blank 20h for unused portions). Unisys will advise the vendor on required drive ID's (Drive model numbers).

FIRMWARE DOWNLOAD FORMAT SPECIFICATION

EXAMPLES: "ST41600N" or "ST41600ND"

4.3 Field Definitions - Non-Mode 5 Based

The following field definitions are defined for drives that cannot download using a single ANSI standard Write Buffer command with Mode 5. The use of the fields in this section is optional. These fields should not be used if the product can operate with the default download sequence that uses a single Write Buffer command and Mode 5.

NOTE TO IMPLEMENTORS OF DOWNLOADING UTILITY:

If using the non-Mode 5 method, the download utility will have to get the Mode Information prior to building the CDB.

MODE Record

This is an ASCII 4 byte identifier for the start of the mode definition section of the header record. If present, it indicates the drive does not support the default download method. The remaining MODE supporting records must be used to determine the appropriate Write Buffer command parameters and sequence.

DATA: "MODE"

Mode-Start

This ASCII 1 byte value indicates what mode of download should be used for all but the last Write Buffer command, in a multiple Write Buffer series to download microcode.

EXAMPLE: "6"

If the drive cannot support multiple Write Buffer commands firmware download and also cannot support the default of a Mode 5 download, then this field shall contain the same value as the Mode-Last field.

Mode-Last

This ASCII 1 byte value indicates what mode of download should be used for the last (or only) Write Buffer command.

EXAMPLE: "7"

Mode Offset Minimum Increment

This 8 byte field identifies the minimum incremental number of decimal bytes (represented in ASCII) of firmware data (offset) the drive will accept for multiple Write Buffer commands. It's format is right justified with leading zeroes.

This value is also used to determine the factor by which an offset greater than the minimum is determined. An offset must be an even multiple of the minimum increment. For example, if the minimum increment is 4096, only offsets in multiples of 4096 (e.g. 8192, 12288, or 65536, etc.) may be used.

FIRMWARE DOWNLOAD FORMAT SPECIFICATION

EXAMPLE: "00004096"

If the drive does not support the use of offsets in the Write Buffer command, this field shall be set to the Size of Firmware Data field value for consistency.

Mode Offset Maximum Increment

This field identifies the maximum incremental number of decimal bytes (represented in ASCII) of firmware data (offset) the drive will accept for multiple Write Buffer commands. It's format is right justified with leading zeroes.

If the drive does not support the use of offsets in the Write Buffer command, this field shall be set to the Size of Firmware Data field value for consistency.

If the drive can accept a single Write Buffer command for download, this field shall contain the Size of Firmware Data field value for consistency.

EXAMPLE: "00305152"

4.4

Firmware Binary Data

This record immediately follows the 8K file header. It contains the actual firmware code. The size of the field is set the value in the Size of Firmware Data field.

FIRMWARE DOWNLOAD FORMAT SPECIFICATION**5. APPENDIX**

This section does not apply to the Supplier that is trying to build a header and binary code file. It is informational only and intended to explain how the information in the header is used.

5.1 Method of Downloading Firmware

The SCSI Write Buffer command shall be used to download and save the firmware on the drive. There are two methods for downloading the firmware. The preferred capability is for the drive to receive the entire firmware file body as a single Write Buffer command. The second method utilizes offsets and modes as defined in the SCSI-3 Write Buffer command to enable the firmware file to be sent in multiple commands.

5.2 Default Download

The following download parameters are defined for backwards compatibility with previous Unisys download utilities, and firmware files. This CDB must be used when the MODE field is not present.

The Write Buffer CDB is set as follows, and the firmware is transferred in a single command:

0x3B	
0x05	05 = Microcode Download
0x00	Buffer ID
0x00	Reserved
0x00	
0x00	
0xXX	Number of bytes sent to drive (3 byte field=Size of Firmware in Hex)
0xXX	
0xXX	
0x00	

5.3 Mode and Offset Controlled Download

A download using the Mode and Offset controls requires the utility to generate a CDB based on the MODE and increment values. Consider the following MODE header information:

FIRMWARE DOWNLOAD FORMAT SPECIFICATION

Field	Value
MODE	MODE
Mode-Start	6
Mode-Last	7
Minimum Incr.	00004096
Maximum Incr.	00305152

Note: For the purposes of this example, the maximum increment was set to the Size of Firmware Data value since the drive is capable of a single Write Buffer command download.

The system for which the download will occur can only send a maximum of 65536 bytes in a SCSI command, despite the drive's ability to receive the firmware in a single command. Since this is the case, the utility will have to generate four CDBs with Mode 6 (the Start value), and another CDB with Mode 7 (Last).

The first 4 CDBs contain the following data:

Table 5-1. Download with Offset, CDB #1

0x3B	
0x06	6 = Download with Offset
0x00	Buffer ID
0x00	Buffer Offset
0x00	
0x00	
0x01	Number of bytes sent to drive (3
0x00	byte field)
0x00	
0x00	

Table 5-2. Download with Offset, CDB #2

0x3B	
0x06	6 = Download with Offset
0x00	Buffer ID
0x01	Buffer Offset
0x00	
0x00	
0x01	Number of bytes sent to drive (3
0x00	byte field)
0x00	
0x00	

FIRMWARE DOWNLOAD FORMAT SPECIFICATION

Table 5-3. Download with Offset, CDB #3

0x3B	
0x06	6 = Download with Offset
0x00	Buffer ID
0x02	Buffer Offset
0x00	
0x00	
0x01	Number of bytes sent to drive (3
0x00	byte field)
0x00	

Table 5-4. Download with Offset, CDB #4

0x3B	
0x06	6 = Download with Offset
0x00	Buffer ID
	Buffer Offset
0x03	
0x00	
0x00	
0x01	Number of bytes sent to drive (3
0x00	byte field)
0x00	
0x00	

The last CDB contains Mode 7, Download with Offset and Save:

Table 5-5. Download with Offset and Save, Last CDB

0x3B	
0x07	7 = Download with Offset and Save
0x00	Buffer ID
0x04	Buffer Offset
0x00	
0x00	
0x00	Number of bytes sent to drive (3
0xAB	byte field)
0x00	
0x00	

5.3 Determining the Method of Download

This section provides a sample algorithm that the utility implementor can use to determine what type of download should be performed based on the information provided in the Header. It is assumed that before this algorithm is processed that the Header data has been validated with the device, i.e. Unisys name in the header, firmware levels compared, and product model compared.

FIRMWARE DOWNLOAD FORMAT SPECIFICATION

DEFINE VARIABLE "SYSCAP" = The maximum length of a SCSI I/O that the system can send for the Write Buffer Command.

IF THE WORD 'MODE' IS NOT PRESENT IN THE HEADER DATA
THEN ; DEFAULT DOWNLOAD METHOD MUST BE USED

IF 'Size of firmware data' > SYSCAP
THEN

ERROR "DOWNLOAD IS NOT POSSIBLE"

ELSE

PERFORM DEFAULT DOWNLOAD (MODE 5, SINGLE COMMAND)

ELSE ; DEFAULT CAN'T BE USED BECAUSE MODE IS PRESENT

IF SYSCAP ≥ 'Mode Offset Maximum Increment'

THEN

IF 'Mode Offset Maximum Increment' = 'Size of firmware data'

THEN

"DO 1 I/O WITH 'Mode Last' IN THE CDB, AND LENGTH =
'Size of firmware data'"

ELSE

"DO 'n' I/Os WITH 'Mode Start' IN THE CDB, AND LENGTH =
'Mode Offset Maximum Increment' UNTIL THE LAST I/O WHICH IS
EXECUTED WITH 'Mode Last' IN THE CDB. DON'T FORGET THAT
THE OFFSETS IN THE CDB HAVE TO INCREMENT WITH EACH I/O."

ELSE ; THIS IS THE CASE WHERE DRIVER FIRMWARE IS BIGGER THAN HOST CAP

IF 'Mode Offset Minimum Increment' > SYSCAP

THEN

ERROR "DOWNLOAD IS NOT POSSIBLE. SYSTEM NOT CAPABLE"

ELSE

"CALCULATE THE LARGEST POSSIBLE BLOCK SIZE THAT IS A
MULTIPLE OF THE 'Mode Offset Minimum Increment', BUT IS LESS
THAN THE SYSCAP SIZE. THEN--

"DO 'n' I/Os WITH 'Mode Start' IN THE CDB, AND LENGTH =
'Newly calculated value < SYSCAP' UNTIL THE LAST I/O WHICH IS
EXECUTED WITH 'Mode Last' IN THE CDB. DON'T FORGET THAT
THE OFFSETS IN THE CDB HAVE TO INCREMENT WITH EACH I/O."

Alanis, Belisario **MV**

From: Leisz, Frank J TR
To: Alanis, Belisario MV; Shelton, Phil MV
Cc: Hale, Wesley MV; Bauman, David A; Stell, Jeffery A
Subject: Test Unit Ready
Date: Friday, September 13, 1996 2:09PM

Hi;

In order to accommodate the SSD request for the addition of the Test Unit Ready command, S/W will continue to use the current interface between the DFAST application and the MCP procedure USERMAINTERQUEST.

To invoke the Test Unit Ready command, the User will pass 11 as the function code. The remaining parameter definitions are the same.

For this function code, the MCP will issued the Test Unit Ready Command (00h) and return any detected error condition in the parameter DEVICERD.

Any problems or comments on this interface?

TIA

SEPT 1996
Information to
Inventor B. Alanis for
Integration with MCP
during DFAST upgrade
development

(E2)

Alanis, Belisario

MV

From: Leisz, Frank J
To: Alanis, Belisario MV
Cc: Houseman, Gloria R; Stell, Jeffery A
Subject: Bootleg 423A MCP (Download Microcode with multi-IO's)
Date: Tuesday, November 19, 1996 1:56PM

Greetings:

I created a new bootleg 423A MCP, which contains the new download function codes and the Ralph's OST patches.

The name of the bootleg MCP is:

(TRANSFER)SYSTEM/423A/MCP/ASDX/UMR ON TRANSFER located on TRPROGD

Hopefully this helps with available machine time, if there are any problems, please call me at sn-385-4487. TIA

Frank

NOV 1996

Information to
Inventor B. Alanis for
Integration with
Unisys Master Control
Program (MCP)

E3

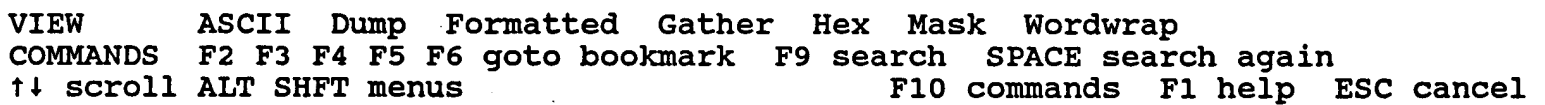
DFAST= ON ENGDATA

File Name	Filekind	Records	Sectors	CreationTime
DFAST	DCALGOLCODE	270	270	08/09/1996
DFAST/NX	DCALGOLCODE	270	270	11/11/1996
DFAST/GAM/DLT/MLTBLK/TUR/NFWCOMPR	DCALGOLCODE	290	297	03/07/1997
DFAST/SPO	DCALGOLCODE	280	288	07/10/1996
DFAST/CHUNK	DCALGOLCODE	270	270	09/23/1996
DFAST/GAMMA/DELTA	DCALGOLCODE	270	270	02/18/1997
DFAST/GAMMA/DELTA/MULTBLK/TUR	DCALGOLCODE	290	297	02/18/1997
DFAST/NXTUR	DCALGOLCODE	290	297	01/28/1997
DFAST/CHUNKY	DCALGOLCODE	286	288	11/27/1996
DFAST/CHUNKY/UNTIL0	DCALGOLCODE	286	288	12/02/1996
DFAST/3DIGIT	DCALGOLCODE	269	270	06/27/1996
DFAST/3DIGIT/SYSTPFORDEV	DCALGOLCODE	270	270	06/27/1996
DFAST/CHUNKY2	DCALGOLCODE	290	297	12/12/1996
DFAST/CHUNKY2/UNISYS	DCALGOLCODE	290	297	01/13/1997
DFAST/NXTURFX	DCALGOLCODE	290	297	01/29/1997

Experimental And
Test Runs by Inventor
B. Alanis during develop-
ments for upgrade to
DFAST,

1996 - 1997

(E4)



Symbol/DFAST/1005000

Header work in
April 1996 by Inventor
working on upgrade to
DFAST

E 5

E 6

U N I S Y S

ENGINEERING INFORMATION RELEASE
SUMMARY SHEETEIR Number Prt Cmp Rv
MV - 200523 001 ARel Date: 19990707
Rev Date:Product Name: CONTRLR SUPPT
Project No : COMMON
Type : CHANGE
Category : SOFTWARE
Eng Affected: MV
Mfg Affected:
REA(s) :RCC: 4
Priority: APage 1 of 23
Summary Shts: 3
Rep/Add Shts: 1
Attachments :

WORKORDER(s): MV44003

Keywrds:

Reason: UPDATE DFAST LVL 1.004 TO LVL 1.005

Explanation - Full:

RELEASES DFAST MEDIA LEVEL 1.005 TO THE PTDTOQUAL KIT AND
RELEASES MEDIA TO POSM. DFAST LEVEL 1.005 ADDS SUPPORT TO THE
NX5820 LVD AND SERVER DOWNLOAD.
34925156-003 TO -004
34925149-003 TO -004EIR Revision: A
Preparer: C. LAKE
Checker :
Init :
Safety :
Authoriz:

Est Date: 19990629

Rel Date: 19990707

EMI Coord:
EMI Info :
Mfg Rev :
Other :
Other :

Authorizer's Verification Statement:

Lo.	Us.	Base/Part	List	Part	Pic	Description	C	BC	No.	Pg.	RM
			Fr/To.	Fr/To.	Fr/To.		C	St	SF.		
3492	4639		F	G	* *	DFASTMT MED ASSY M42		PL		2	4
3492	4639-003			F	G	DFASTMT MED ASSY M42		C	I		
3492	4639-004				G	DFASTMT MED ASSY M42	1.0	N	A		
3492	5149		F	G	* *	DFASTQIC MED ASSY M17		PL		2	6
3492	5149-003			F	G	DFASTQIC MED ASSY M17		C	I		
3492	5149-004				G	DFASTQIC MED ASSY M17	1.0	N	A		
3492	5156		F	G	* *	DFASTHIC MED ASSY M48		PL		2	8
3492	5156-003			F	G	DFASTHIC MED ASSY M48	LV 1.0	C	I		
3492	5156-004				G	DFASTHIC MED ASSY M48	LV 1.0	N	A		
3493	0222		E	G	* *	DFASTHIC MSTR MED M48		TL		2	10
3493	0222-003			E	G	DFASTHIC MSTR MED M48		C	I		
3493	0222-004				G	DFASTHIC MSTR MED M48		N	A		
3493	0230		E	G	* *	DFASTHIC LAB TXT M48		TL		2	12
3493	0230-003			E	G	DFASTHIC LAB TXT M48		C	I		
3493	0230-004				G	DFASTHIC LAB TXT M48		N	A		
3493	1014		E	F	* *	DFASTQIC MSTR MED M17		TL		2	14
3493	1014-003			E	F	DFASTQIC MSTR MED M17		C	I		
3493	1014-004				F	DFASTQIC MSTR MED M17		N	A		
3493	1022		E	F	* *	DFASTQIC LAB TXT M17		TL		2	16
3493	1022-003			E	F	DFASTQIC LAB TXT M17		C	I		
3493	1022-004				F	DFASTQIC LAB TXT M17		N	A		

EXHIBIT

E-8

ENGINEERING INFORMATION RELEASE SUMMARY SHEET

EIR Number	Prt	Cmp	Rv
MV - 200523	001		A

Rel Date: 19990707
Rev Date:

Product Name: CONTRLR SUPPT
RCC: 4 Priority: A

Page 2 of 23

Lo.	List	Part	Pic			C	BC	No.	Pg.
Us.	Base/Part	Fr/To.	Fr/To.	Fr/To.	Description	C	St SF.		RM
3493	1030	E	F	*	* DFASMT MSTR MED M42		TL	2	18
3493	1030-003		E	F	DFASMT MSTR MED M42	C	I		
3493	1030-004			F	DFASMT MSTR MED M42	N	A		
3493	1048	E	F	*	* DFASMT LAB TXT M42		TL	2	20
3493	1048-003		E	F	DFASMT LAB TXT M42	C	I		
3493	1048-004			F	DFASMT LAB TXT M42	N	A		

U N I S Y S

ENGINEERING INFORMATION RELEASE
SUMMARY SHEET
CONTINUATION

EIR Number Prt Cmp Rv
MV - 200523 001 A

Rel Date: 19990707
Rev Date:

Product Name: CONTRLR SUPPT
RCC: 4 Priority: A

Page 3 of 23

EIR CONTINUATION REV: A
RELEASES DFAST MEDIA LEVEL 1.005 TO THE PTDTOQUAL KIT AND
RELEASES MEDIA TO POSM. DFAST LEVEL 1.005 ADDS SUPPORT TO THE
NX5820 LVD AND SERVER DOWNLOAD.
34925156-003 TO -004
34925149-003 TO -004
34924639-003 TO -004
SCRAP OLD PART NUMBERS.

Base Number	List Fr/To	Pic Fr/To	Description	BM	Pr	RCC	No Sh
3492 4639	F G	* *	DFASTMT MED ASSY M42	PL	A	4	1

CEDB CHANGES:

BASE NUMBER RELATED CHANGES:

	From	To
CHG Range:	000-003	000-004
CHG EPC:	*	COMMON
ADD PFC:		SOFTWR

PART NUMBER RELATED CHANGES:

	Part Fr/To	Description	CC	St
3492 4639-003	F G	DFASTMT MED ASSY M42	C	I

	From	To
CHG Desc:	DFASTMT LV: 1.004 ..	DFASTMT MED ASSY M42
CHG Status:	A	I
ADD Newpn:		3492 4639-004

DIFFERENCES BETWEEN 3492 4639-003 REPLACED-INACTIVE BY 3492 4639-004

	Find Ch	Alt From	To
CHG	1	Part: 3937 6975-000 MAGNETIC TAPE, 1200 FT. BLANK MEDIA ASSY	3937 7007-000 MAG TAPE 3600 FOOT
DEL R001		Ref: 3493 1030-003 DFASTMT MSTR MED M42 DFASTMT MSTR MED M42	
CHG	R002	Ref: 3493 1048-003 DFASTMT LAB TXT M42 DFASTMT LAB TXT M42	3493 1030-004 DFASTMT MSTR MED M42 DFASTMT MSTR MED M42

Base Number	List Fr/To	Pic Fr/To	Description	BM	Pr	RCC	No Sh
3492 4639	F G	* *	DFASTMT MED ASSY M42	PL	A	4	2

DIFFERENCE REPORT Continued

Find Ch	Alt From	To
ADD R003	Ref:	3493 1048-004 DFASTMT LAB TXT M42 DFASTMT LAB TXT M42

PART NUMBER RELATED CHANGES:

Part Fr/To	Description	CC	St
3492 4639-004	G DFASTMT MED ASSY M42	1.005	N A

From	To
ADD Desc:	DFASTMT MED ASSY M42
ADD Status:	A

MANUFACTURING STATUS INFORMATION

DISPOSITION OF PARTS:

_____ SCRAP OLD PARTS	_____ DEplete OLD STOCK	_____ REWORK
_____ INCORP ON OPEN ORDERS	_____ DRAW MATL FOR EVALUATN	
_____ USE UNTIL NEW PARTS AVAIL	_____ INCORP NEXT TIME BLD	_____ OTHER

IMPLEMENT ON ASSEMBLIES/PARTS:

_____ SERIAL NUMBER:	
_____ REWORK:	_____ IN MFG _____ IN TEST _____ AT VENDOR _____ IN FIELD

COMMENTS:

Base Number	List Fr/To	Pic Fr/To	Description	BM	Pr	RCC	No Sh
3492 5149	F G	* *	DFASTQIC MED ASSY M17	PL	A	4	1

CEDB CHANGES:

BASE NUMBER RELATED CHANGES:

	From	To
CHG Range:	000-003	000-004
CHG EPC:	*	COMMON
ADD PFC:		SOFTWR

PART NUMBER RELATED CHANGES:

	Part Fr/To	Description	CC	St
3492 5149-003	F G	DFASTQIC MED ASSY M17	C	I

	From	To
CHG Desc:	DFASTQIC LV: 1.004 ..	DFASTQIC MED ASSY M17
CHG Status:	A	I
ADD Newpn:		3492 5149-004

DIFFERENCES BETWEEN 3492 5149-003 REPLACED-INACTIVE BY 3492 5149-004

	Find Ch	Alt	From	To
DEL R001				
	Ref:		3493 1014-003	
			DFASTQIC MSTR MED M17	
			DFASTQIC MSTR MED M17	
R002				
CHG	Ref:		3493 1022-003	3493 1014-004
			DFASTQIC LAB TXT M17	DFASTQIC MSTR MED M17
			DFASTQIC LAB TXT M17	DFASTQIC MSTR MED M17
ADD R003				
	Ref:		3493 1022-004	
			DFASTQIC LAB TXT M17	
			DFASTQIC LAB TXT M17	

Base Number	List Fr/To	Pic Fr/To	Description	BM	Pr	RCC	No Sh
3492 5149	F G	* *	DFASTQIC MED ASSY M17	PL	A	4	2

PART NUMBER RELATED CHANGES:

	Part Fr/To	Description	CC	St
3492 5149-004	G	DFASTQIC MED ASSY M17 1.005	N	A

	From	To
ADD Desc:		DFASTQIC MED ASSY M17 1.005
ADD Status:		A

MANUFACTURING STATUS INFORMATION

DISPOSITION OF PARTS:

_____ SCRAP OLD PARTS	_____ DEplete OLD STOCK	_____ Rework
_____ INCORP ON OPEN ORDERS	_____ DRAW MATL FOR EVALUATN	
_____ USE UNTIL NEW PARTS AVAIL	_____ INCORP NEXT TIME BLD	_____ OTHER

IMPLEMENT ON ASSEMBLIES/PARTS:

_____ SERIAL NUMBER:	
_____ Rework:	_____ IN MFG _____ IN TEST _____ AT VENDOR _____ IN FIELD

COMMENTS:

Base Number	List Fr/To	Pic Fr/To	Description	BM	Pr	RCC	No Sh
3492 5156	F G	* *	DFASTHIC MED ASSY M48	PL	A	4	1

CEDB CHANGES:

BASE NUMBER RELATED CHANGES:

	From	To
CHG Range:	000-003	000-004
CHG EPC:	*	COMMON
ADD PFC:		SOFTWR

PART NUMBER RELATED CHANGES:

	Part Fr/To	Description	CC	St
3492 5156-003	F G	DFASTHIC MED ASSY M48 LV 1.00	C	I

	From	To
CHG Desc:	DFASTHIC LV:1.004 ..	DFASTHIC MED ASSY M48 LV 1.00
CHG Status:	A	I
ADD Newpn:		3492 5156-004

DIFFERENCES BETWEEN 3492 5156-003 REPLACED-INACTIVE BY 3492 5156-004

	Find Ch	Alt From	To
DEL R001			
	Ref:	3493 0222-003	
		DFASTHIC MSTR MED M48	
		DFASTHIC MSTR MED M48	
CHG R002			
	Ref:	3493 0230-003	3493 0222-004
		DFASTHIC LAB TXT M48	DFASTHIC MSTR MED M48
		DFASTHIC LAB TXT M48	DFASTHIC MSTR MED M48
ADD R003			
	Ref:	3493 0230-004	
		DFASTHIC LAB TXT M48	
		DFASTHIC LAB TXT M48	

Base Number	List Fr/To	Pic Fr/To	Description	BM	Pr	RCC	No Sh
3492 5156	F G	* *	DFASTHIC MED ASSY M48	PL	A	4	2

PART NUMBER RELATED CHANGES:

	Part Fr/To	Description	CC	St
3492 5156-004	G	DFASTHIC MED ASSY M48 LV 1.005	N	A

	From	To
ADD Desc:		DFASTHIC MED ASSY M48 LV 1.005
ADD Status:		A

MANUFACTURING STATUS INFORMATION

DISPOSITION OF PARTS:

_____ SCRAP OLD PARTS	_____ DEplete OLD STOCK	_____ REWORK
_____ INCORP ON OPEN ORDERS	_____ DRAW MATL FOR EVALUATN	
_____ USE UNTIL NEW PARTS AVAIL	_____ INCORP NEXT TIME BLD	_____ OTHER

IMPLEMENT ON ASSEMBLIES/PARTS:

_____ SERIAL NUMBER:	
_____ REWORK:	_____ IN MFG _____ IN TEST _____ AT VENDOR _____ IN FIELD

COMMENTS:

Base Number	List Fr/To	Pic Fr/To	Description	BM	Pr	RCC	No Sh
3493 0222	E G	* *	DFASTHIC MSTR MED M48	TL	A	4	1

CEDB CHANGES:

BASE NUMBER RELATED CHANGES:

	From	To
CHG Range:	000-003	000-004
CHG EPC:	*	COMMON
ADD PFC:		SOFTWR

PART NUMBER RELATED CHANGES:

	Part Fr/To	Description	CC	St
3493 0222-003	E G	DFASTHIC MSTR MED M48	C	I

	From	To
ADD Desc:		DFASTHIC MSTR MED M48
CHG Status:	A	I
ADD Newpn:		3493 0222-004

	Part Fr/To	Description	CC	St
3493 0222-004	G	DFASTHIC MSTR MED M48	N	A

	From	To
ADD Desc:		DFASTHIC MSTR MED M48
ADD Status:		A

Base Number	List Fr/To	Pic Fr/To	Description	BM	Pr	RCC	No Sh
3493 0222	E G	* *	DFASTHIC MSTR MED M48	TL	A	4	2

MANUFACTURING STATUS INFORMATION

DISPOSITION OF PARTS:

_____ SCRAP OLD PARTS	_____ DEplete OLD STOCK	_____ REWORK
_____ INCORP ON OPEN ORDERS	_____ DRAW MATL FOR EVALUATN	
_____ USE UNTIL NEW PARTS AVAIL	_____ INCORP NEXT TIME BLD	_____ OTHER

IMPLEMENT ON ASSEMBLIES/PARTS:

_____ SERIAL NUMBER: _____
_____ REWORK: _____ IN MFG _____ IN TEST _____ AT VENDOR _____ IN FIELD

COMMENTS:

Base Number	List Fr/To	Pic Fr/To	Description	BM	Pr	RCC	No Sh
3493 0230	E G	* *	DFASTHIC LAB TXT M48	TL	A	4	1

CEDB CHANGES:

BASE NUMBER RELATED CHANGES:

	From	To
CHG Range:	000-003	000-004
CHG EPC:	*	COMMON
ADD PFC:		LABEL

PART NUMBER RELATED CHANGES:

	Part Fr/To	Description	CC	St
3493 0230-003	E G	DFASTHIC LAB TXT M48	C	I

	From	To
ADD Desc:		DFASTHIC LAB TXT M48
CHG Status:	A	I
ADD Newpn:		3493 0230-004

	Part Fr/To	Description	CC	St
3493 0230-004	G	DFASTHIC LAB TXT M48	N	A

	From	To
ADD Desc:		DFASTHIC LAB TXT M48
ADD Status:		A

Base Number	List Fr/To	Pic Fr/To	Description	BM	Pr	RCC	No Sh
3493 0230	E G	* *	DFASTHIC LAB TXT M48	TL	A	4	2

MANUFACTURING STATUS INFORMATION

DISPOSITION OF PARTS:

_____ SCRAP OLD PARTS	_____ DEplete OLD STOCK	_____ REWORK
_____ INCORP ON OPEN ORDERS	_____ DRAW MATL FOR EVALUATN	
_____ USE UNTIL NEW PARTS AVAIL	_____ INCORP NEXT TIME BLD	_____ OTHER

IMPLEMENT ON ASSEMBLIES/PARTS:

_____ SERIAL NUMBER: _____
_____ REWORK: _____ IN MFG _____ IN TEST _____ AT VENDOR _____ IN FIELD

COMMENTS:

Base Number	List Fr/To	Pic Fr/To	Description	BM	Pr	RCC	No Sh
3493 1014	E F	* *	DFASTQIC MSTR MED M17	TL	A	4	1

CEDB CHANGES:

BASE NUMBER RELATED CHANGES:

	From	To
CHG Range:	000-003	000-004
ADD Med Cd:		CT
CHG EPC:	*	COMMON
ADD PFC:		SOFTWR

PART NUMBER RELATED CHANGES:

	Part Fr/To	Description	CC	St
3493 1014-003	E F	DFASTQIC MSTR MED M17	C	I

	From	To
ADD Desc:		DFASTQIC MSTR MED M17
CHG Status:	A	I
ADD Newpn:		3493 1014-004

	Part Fr/To	Description	CC	St
3493 1014-004	F	DFASTQIC MSTR MED M17	N	A

	From	To
ADD Desc:		DFASTQIC MSTR MED M17
ADD Status:		A

U N I S Y S

ENGINEERING INFORMATION RELEASE
DETAIL SHEETEIR Number Prt Cmp Rv
MV - 200523 001 A
Page 15 of 23

Base Number	List Fr/To	Pic Fr/To	Description	BM	Pr	RCC	No Sh
3493 1014	E F	* *	DFASTQIC MSTR MED M17	TL	A	4	2

MANUFACTURING STATUS INFORMATION

DISPOSITION OF PARTS:

_____ SCRAP OLD PARTS	_____ DEplete OLD STOCK	_____ REWORK
_____ INCORP ON OPEN ORDERS	_____ DRAW MATL FOR EVALUATN	
_____ USE UNTIL NEW PARTS AVAIL	_____ INCORP NEXT TIME BLD	_____ OTHER

IMPLEMENT ON ASSEMBLIES/PARTS:

_____ SERIAL NUMBER: _____
_____ REWORK: _____ IN MFG _____ IN TEST _____ AT VENDOR _____ IN FIELD

COMMENTS:

Base Number	List Fr/To	Pic Fr/To	Description	BM	Pr	RCC	No Sh
3493 1022	E F	* *	DFASTQIC LAB TXT M17	TL	A	4	1

CEDB CHANGES:

BASE NUMBER RELATED CHANGES:

	From	To
CHG Range:	000-003	000-004
ADD Med Cd:		CT
CHG EPC:	*	COMMON
ADD PFC:		LABEL

PART NUMBER RELATED CHANGES:

	Part Fr/To	Description	CC	St
3493 1022-003	E F	DFASTQIC LAB TXT M17	C	I

	From	To
ADD Desc:		DFASTQIC LAB TXT M17
CHG Status:	A	I
ADD Newpn:		3493 1022-004

	Part Fr/To	Description	CC	St
3493 1022-004	F	DFASTQIC LAB TXT M17	N	A

	From	To
ADD Desc:		DFASTQIC LAB TXT M17
ADD Status:		A

Base Number	List Fr/To	Pic Fr/To	Description	BM	Pr	RCC	No Sh
3493 1022	E F	* *	DFASTQIC LAB TXT M17	TL	A	4	2

MANUFACTURING STATUS INFORMATION

DISPOSITION OF PARTS:

_____ SCRAP OLD PARTS	_____ DEplete OLD STOCK	_____ REWORK
_____ INCORP ON OPEN ORDERS	_____ DRAW MATL FOR EVALUATN	
_____ USE UNTIL NEW PARTS AVAIL	_____ INCORP NEXT TIME BLD	_____ OTHER

IMPLEMENT ON ASSEMBLIES/PARTS:

_____ SERIAL NUMBER: _____
_____ REWORK: _____ IN MFG _____ IN TEST _____ AT VENDOR _____ IN FIELD

COMMENTS:

Base Number	List Fr/To	Pic Fr/To	Description	BM	Pr	RCC	No Sh
3493 1030	E F	* *	DFASTMT MSTR MED M42	TL	A	4	1

CEDB CHANGES:

BASE NUMBER RELATED CHANGES:

	From	To
CHG Range:	000-003	000-004
CHG EPC:	*	COMMON
ADD PFC:		SOFTWR

PART NUMBER RELATED CHANGES:

	Part Fr/To	Description	CC	St
3493 1030-003	E F	DFASTMT MSTR MED M42	C	I

	From	To
ADD Desc:		DFASTMT MSTR MED M42
CHG Status:	A	I
ADD Newpn:		3493 1030-004

	Part Fr/To	Description	CC	St
3493 1030-004	F	DFASTMT MSTR MED M42	N	A

	From	To
ADD Desc:		DFASTMT MSTR MED M42
ADD Status:		A

Base Number	List Fr/To	Pic Fr/To	Description	BM	Pr	RCC	No Sh
3493 1030	E F	* *	DFASTMT MSTR MED M42	TL	A	4	2

MANUFACTURING STATUS INFORMATION

DISPOSITION OF PARTS:

<input type="checkbox"/> SCRAP OLD PARTS	<input type="checkbox"/> DEplete OLD STOCK	<input type="checkbox"/> REWORK
<input type="checkbox"/> INCORP ON OPEN ORDERS	<input type="checkbox"/> DRAW MATL FOR EVALUATN	
<input type="checkbox"/> USE UNTIL NEW PARTS AVAIL	<input type="checkbox"/> INCORP NEXT TIME BLD	<input type="checkbox"/> OTHER

IMPLEMENT ON ASSEMBLIES/PARTS:

<input type="checkbox"/> SERIAL NUMBER:	
<input type="checkbox"/> REWORK:	<input type="checkbox"/> IN MFG <input type="checkbox"/> IN TEST <input type="checkbox"/> AT VENDOR <input type="checkbox"/> IN FIELD

COMMENTS:

Base Number	List Fr/To	Pic Fr/To	Description	BM	Pr	RCC	No Sh
3493 1048	E F	* *	DFASTMT LAB TXT M42	TL	A	4	1

CEDB CHANGES:

BASE NUMBER RELATED CHANGES:

	From	To
CHG Range:	000-003	000-004
ADD Med Cd:		MT
CHG EPC:	*	COMMON
ADD PFC:		LABEL

PART NUMBER RELATED CHANGES:

	Part Fr/To	Description	CC	St
3493 1048-003	E F	DFASTMT LAB TXT M42	C	I

	From	To
ADD Desc:		DFASTMT LAB TXT M42
CHG Status:	A	I
ADD Newpn:		3493 1048-004

	Part Fr/To	Description	CC	St
3493 1048-004	F	DFASTMT LAB TXT M42	N	A

	From	To
ADD Desc:		DFASTMT LAB TXT M42
ADD Status:		A

Base Number	List Fr/To	Pic Fr/To	Description	BM	Pr	RCC	No Sh
3493 1048	E F	* *	DFASTMT LAB TXT M42	TL	A	4	2

MANUFACTURING STATUS INFORMATION

DISPOSITION OF PARTS:

<input type="checkbox"/> SCRAP OLD PARTS	<input type="checkbox"/> DEplete OLD STOCK	<input type="checkbox"/> REWORK
<input type="checkbox"/> INCORP ON OPEN ORDERS	<input type="checkbox"/> DRAW MATL FOR EVALUATN	
<input type="checkbox"/> USE UNTIL NEW PARTS AVAIL	<input type="checkbox"/> INCORP NEXT TIME BLD	<input type="checkbox"/> OTHER

IMPLEMENT ON ASSEMBLIES/PARTS:

<input type="checkbox"/> SERIAL NUMBER:	
<input type="checkbox"/> REWORK:	<input type="checkbox"/> IN MFG <input type="checkbox"/> IN TEST <input type="checkbox"/> AT VENDOR <input type="checkbox"/> IN FIELD

COMMENTS:

U N I S Y S

ENGINEERING INFORMATION RELEASE
REPORTS/ADDENDUM

EIR Number Prt Cmp Rv
MV - 200523 001 A

Rel Date: 19990707
Rev Date:

Product Name: CONTRLR SUPPT
RCC: 4 Priority: A

Page 22 of 23

TOP UNITS AFFECTED:

Top Unit Part	Sta	Bom	DCL	Description
2346 1809-999	A	SU	MV	KIT: PTDTOQUAL
3445 3068-000	I	SU	MV	KIT,SCSI FIRMWARE DOWNLOAD
3445 3068-001	I	SU	MV	KIT,SCSI FIRMWARE DOWNLOAD
3445 3068-002	I	SU	MV	KIT,SCSI FIRMWARE DOWNLOAD
3445 3068-003	A	SU	MV	KIT,SCSI FIRMWARE DOWNLOAD
4490 2542-000	A	SU	MV	KIT, FCN-USR4000-010 A-MT
4490 2567-000	A	SU	MV	INDEX, FCN USR4000-010

3492 4639-003 DFASTMT MED ASSY M42

DFASTMT MED ASSY M42

DIFFERENCES BETWEEN 3492 4639-003 REPLACED-INACTIVE BY 3492 4639-004

	Find	Ch	Alt	From	To
	----	---	---	-----	---
CHG		1		3937 6975-000	3937 7007-000
DEL Ref	R001			3493 1030-003	
CHG Ref	R002			3493 1048-003	3493 1030-004
ADD Ref	R003				3493 1048-004

3492 5149-003 DFASTQIC MED ASSY M17

DFASTQIC MED ASSY M17

DIFFERENCES BETWEEN 3492 5149-003 REPLACED-INACTIVE BY 3492 5149-004

	Find	Ch	Alt	From	To
	----	---	---	-----	---
DEL Ref	R001			3493 1014-003	
CHG Ref	R002			3493 1022-003	3493 1014-004
ADD Ref	R003				3493 1022-004

3492 5156-003 DFASTHIC MED ASSY M48

DFASTHIC MED ASSY M48 LV 1.00

DIFFERENCES BETWEEN 3492 5156-003 REPLACED-INACTIVE BY 3492 5156-004

	Find	Ch	Alt	From	To
	----	---	---	-----	---
DEL Ref	R001			3493 0222-003	
CHG Ref	R002			3493 0230-003	3493 0222-004
ADD Ref	R003				3493 0230-004

EXHIBIT

E-9

```

%% VERSION 90.032.001
%% SET STACK LIMIT 100
%% RESET LIST XREF
%% SET VERSION 01.005.000
%% SET OMIT
% PATCHFOR DOWNLOAD IN MULTIPLE CHUNKS
#####00000092
00010000
00011000
00012000
00013000
00013500010040006
00014000
%
% Class Unisys 00015000
% 00016000
% 00017000
% This material is restricted and proprietary to the 00018000
% ##### Unisys Corporation and is not to be reproduced, shown, 00019000
% ## or disclosed outside the Unisys Corporation. Customer 00020000
% ## Services Engineering restricted and proprietary data 00021000
% ## is furnished solely for use by Unisys personnel in 00022000
% ## servicing customer's equipment. 00023000
% ## 00024000
% ## 00025000
% ## This document is the property of and shall be returned 00026000
% ## to Unisys Corporation, One Unisys Place, Detroit, MI 00027000
% ##### 48232. 00028000
% 00029000
% 00030000
% Material Copyright (C) 1994. 00031000
% 00032000
% 00033000
% 00034000
% 00035000
#####00036000
%% PAGE 00037000
#####00038000
%
% TARGET / F W L O A D 00039000
% 00040000
% 00041000
% Patch History 00042000
% 00043000
#####00044000
4/94 Initial version. Will download A-code files to SBC controllers 00045000
and SCSI Disk Drives. 00046000
4/94 Changes - Qualification phase for official CSPO release 01.001. 00047000010010001
5/94 Add 'Express Mode' for engineering (DISKS only) release 01.002. 00047200010020003
6/94 Change location of some display statements. --- release 01.002. 00047500010020002
10/94 Fixed seg array error in Verifyfile procedure. release 01.002. 00047600010020004
#####00048000
00049000
DOWNLOAD FIRMWARE to A-SERIES TARGETS (DFAST) 00052000
===== 00053000
00054000
Utility's Part Number: 3492 4639 00055000
00056000
This utility's primary function is to load microcode to SCSI BUS 00057000
CONTROLLERS (SBC) and/or SCSI disk drives, and must be marked as a 00058000
PPed (Privileged Program) in order to operate. 00059000
00060000
Interaction with the user is via prompts at either a terminal or 00061000
an ODT. Terminal prompts use regular I/O. ODT inputs use the 00062000
ACCEPT statement. Otherwise the input rules are the same. The 00063000
program requires interactive user participation to execute properly. 00064000
00065000
00066000
Load: 00067000
----- 00068000
00069000
To operate, two basic elements must be present: an SBC controller or 00070000
SCSI disk drive, and a microcode file on disk or tape. The microcode 00071000
must reside on a unit served by a different controller or disk than the 00072000
one being initialized. If a "critical unit" exists on the string served 00073000
by the controller being initialized and there is only one path to that 00074000
critical unit, the system will reject the attempt to download the 00075000
microcode. 00076000
00077000

```

DI

```

1) Step one is to determine the code file's capability by having the user enter a file name. Normal Family Substitution rules are in effect. If the file cannot be found, the user is prompted to enter another file name. The file name may include an "ON <family>" in the file declaration if it resides on disk (e.g., (UCODE)XYZ/123 ON MYPACK). If the file resides on tape, the file name only would be entered (e.g., SCZFRM).

If the code file is not a valid SBC or disk drive Acode file, the file is rejected and the user is prompted to enter another file name.

2) Step two is to determine the SBC controller or drive capability. The user enters a SBC or drive number and the utility reads the unit's attributes. If the SBC or drive number does not represent the correct target, or if the target attributes do not match those of the Acode file, the user is prompted to enter a new target number or 'Quit'. The SBC controller or drive must be reserved.

>>>> There may be no way to prove that the SBC controller controls any
>>>> given drive or that it is the only path to the drive or that
>>>> another path exists! The User should do an "OL" on the ODT
>>>> to verify the paths available for the target.

Verifyfile:
-----

Verify is simpler than Load since no SBC or drive is involved. The user answers the file name prompt and the Verify routine used by Load is called to generate a report on the attributes associated with that file.

Verify allows the user to cycle through multiple, potential A-code files until 'Quit' is entered.
$$ POP OMIT
BEGIN
*****
%           Structure Generating Declarations
*****
DEFINE
    OVHD          = 12 #           % At front of each code segment
    ,XSTATBYTES   = 254 #          % Read Unit Status return length
    ,EIGHTK       = 8192 #
    ,ENDSGD       = #
    ,SYSCAP       = 393216 #
    ,MAXELEMENTS  = 1 #           % MAX ELEMENT PER DIMENSION
    ,MAXROWS      = 48 #          % MAX NUMBER OF ROWS
;

FILE
    CODE( KIND     = TAPE,         % UNKNOWN TYPE
    ***** LABELKIND = UNLABELLED, % READ ONLY
    FILEUSE       = IN,
    OPTIONAL      = TRUE,
    NEWFILE       = FALSE,
    DEPENDENTSPECS = TRUE)
    ,LINE( KIND     = PRINTER,
    FILEUSE       = OUT,
    FRAME SIZE    = 8,
    MAXRECSIZE    = 132)
    ,RMT( KIND     = REMOTE,
    FILEUSE       = IO,
    BLOCKSTRUCTURE = EXTERNAL,
    FRAME SIZE    = 8,
    MAXRECSIZE    = 132)
;

DIM. DIRECT EBCDIC ARRAY
IFFER IMLBUF2 [0:0,0:0] % To be resized
      IMLBUF3 [0:0,0:0] % To be resized
      IMLBUF  [0:0]
;

```

```

00078000
00079000
00080000
00081000
00082000
00083000
00084000
00085000
00086000
00087000
00088000
00089000
00090000
00091000
00092000
00093000
00094000
00095000
00096000
00097000
00098000
00099000
00100000
00101000
00102000010020003
00103000010020003
00104000
00105000
00106000
00107000
00108000
00109000
00110000
00111000
00112000
00113000
00114000
00115000
00116000
00117000
00118000
00119000
00120000
00120500010040006
0012060001.005.000
0012080001.005.000
00121000
00122000
00123000
00124000
00125000
00126000
00127000
00128000
00129000
00130000
00131000
00132000
00133000
00134000
00135000
00136000
00137000
00138000
00139000
00140000
00141000
00142000010040006
0014220001.005.001
00142500010040006
00143000

```

2 DIMENSIONAL
BUFFER

```

EBCDIC ARRAY
  CODEREC      [0:0]          % CODE record          00144000
  ,ETSTMP      [0:23]        % Timestamp data        00145000
  ,FCUSTID     [0:7]         % FW file customer ID   00146000
  ,FNEWFWLEVEL [0:3]         % FW file new firmware level 00147000
  ,FVENDORID   [0:7]         % FW file vendor ID     00148000
  ,OLDFWLEVEL  [0:3]         % FW level before download 00149000
  ,HDR         [0:0]         % CODE record zero      00150000
  ,HDPRESULT   [0:29]        %                       00150500010020003
  ,INBUF       [0:131]      % User input buffer      00151000
  ,L           [0:131]      % Formatting space       00152000
  ,SCR2        [0:131]      % Scratch 2              00153000
  ,SCR         [0:131]      % Scratch space           00154000
  ,TTL         [0:89]       % ASCII to EBCDIC conversion 00154500010040006
  ,SHORTBUF    [0:XSTATBYTES] % CODE.TITLE           00155000
  ,ESHORTBUF   [0:XSTATBYTES] % UMR Interface calls  00156000
  ,SAVEFWLVL   [0:7]        % ASCII to EBCDIC conversion 00157000
  ,OEM         [0:71]       % FW file data area of FW level 00158000
  ;                                           00159000
                                           00159500010020004
                                           00160000
REAL ARRAY
  RCODE[0]      = CODEREC          00161000
  ,RHDR[0]      = HDR              00162000
  ,TSTMP[0]     = ETSTMP           00163000
  ,X[0:0]       00164000
  ;                                           00165000
                                           00165500010020004
                                           00166000
                                           00167000
HEX ARRAY
  HSCR[0] = SCR;                    00168000
                                           00169000
                                           00170000
POINTER
  DUMMY          % Never used        00171000
  ,IMLP          % To IMLBUF         00172000
  ,IMLP2         % To IMLBUF2        00173000
  ,IMLP3         % To IMLBUF3        0017320001.005.000
  ,PC            % To CODEREC        0017340001.005.000
  ,PL            % To L              00174000
  ;                                           00174000
                                           00175000
                                           00176000
                                           00177000
PICTURE
  DATETIME(N/99I99I99" @ "N:99I99); 00178000
                                           00179000
                                           00180000
TRUTHSET\
  NUMBERS ("0123456789")             00181000
  ,HEXDIGITS (NUMBERS OR "ABCDEF")    00181200010020004
  ,ENDER (" .");                      00181400010020004
                                           00182000010020004
                                           00183000
TRANSLATETABLE
  UPCASER(EBCDIC TO EBCDIC,          00184000
    "abcdefghijklmnopqrstuvwxyz" TO 00185000
    "ABCDEFGHIJKLMNOPQRSTUVWXYZ");    00186000
                                           00187000
                                           00188000
#####00189000
% Real and Integer Variables          00190000
#####00191000
REAL
  % These Reals are either bit buckets or other values which 00192000
  % would not survive integerization. 00193000
  FKAIW          % FILEKIND Attribute Info Word             00194000
  ,MRD           % Maintenance Result Descriptor bits       00195000
  ,SYSTEMTYPE    % All, A14, A16, A18, A19 are valid        00196000
  ,VAL           % Value of hex to decimal conversion       001970000010010001
  ;                                           00197500010020004
                                           00198000
                                           00199000
INTEGER
  CODEEOF        % CODE.LASTRECORD                          00200000
  ,CODEMRS       % CODE.MAXRECSIZE                          00201000
  ,CTLUNIT       % PCD unit number of CTL                   00202000
  ,FHDRBYTES     % # bytes in header record                 00203000
  ,FCODEBYTES    % Code size (# bytes) in firmware file     00204000
  ;                                           00205000

```

```

,FUNCTION          % Operating mode for this run          00206000
,LL                % Number of bytes remaining in PL      00207000
,SEGINX            % Segment starting record index        00208000
,SEGMRS            % Segment maxrecsize                   00209000
,SOFAR             % Bytes moved to IMLBUF so far          00210000
,TIMER             % Drop dead timer (intervals)           00211000
,NUMREPEWLVLVS     % Number of replaceable firmware levels 00212000
,DRIVEIDLOC        % Array index to FW drive IDs           00213000
,NUMDRIVEID        % Number of drive IDs valid for the code 00214000
,OPTODO            % Opcode parameter to UMR               00215000
,NUMBROFIOS        % # of IO's to do for multi chunk dwnld 00215200010040006
,SIZEOFLLSTIO      % Size of last IO to do multi chunk dnlld 00215400010040006
,NROWS             % Number of rows required for download  00215500010040006
,MROWS             % Number of MOD rows as residual rows    0021560001.005.000
;                                                          00216000010040006
;                                                          00217000
;*****00218000
% Boolean Variables 00219000
;*****00220000
BOOLEAN 00221000
,CTLASSIGNED      % CTL assigned 00222000
,SDMASSIGNED      % SCSI disk drive assigned 00223000
,DEBUG            % Internal debug mode 00224000
,ODTMODE          % User is at a console or terminal 00225000
,ATODT            % User is at console 00226000
,SKIPODTPRINT     % Don't print ODT display 00226500010010001
,EXPRESSMODE      % Omit many displays (for engineering) 00226600010020003
,VERIFYONLY       % Don't build FW image in buffer 00226800010020004
,FIRSTTIME        % 0022690001.005.000
; 00227000
;*****00228000
;*****00229000
% Structure Independent Defines 00230000
;*****00231000
DEFINE 00232000
,ADD              = REPLACE PL:PL BY # 00233000
,B                = BOOLEAN # 00234000
,BLANK            = REPLACE L BY " " FOR 22 WORDS # 00235000
,C                = , # 00236000
,CLEARINBUF       = REPLACE INBUF BY " " FOR 12 WORDS # 00237000
,NUL              = 48"00" # 00238000
,P                = POINTER # 00239000
,PUT(N)           = REPLACE PL:L[N] BY # 00240000
,SHOW(S,TF)       = BEGIN 00241000
                   START S; 00242000
                   SHOWIT(TF); 00243000
                   END # 00244000
,START            = REPLACE PL:L BY # 00245000
;*****00246000
% Formatting constants 00247000
;*****00248000
,TXTLOC30         = 30 # 00249000
,TXTLOC22         = 22 # 00250000
,TXTLOC15         = 15 # 00251000
,ERRLOC           = (TXTLOC30+2) # 00252000
;*****00253000
% Function values 00254000
;*****00255000
,LOADCTLV         = 1 # 00256000
,LOADEVV          = 2 # 0025700001.005.000
,VERIFYV          = 3 # 00258000
;*****00259000
% Code File Header (most recent version) 00260000
% Elements for all fields include their location and their length. 00261000
;*****00262000
,CUSTIDLOC        = 0 # 00263000
,CUSTIDLNG        = 8 # 00264000
,CUSTIDEXP        = "FUJITSU " # 00264500010040006
,CUSTIDEXP        = "UNISYS " # 00265000010040006
,HDRECSIZELOC     = 8 # 00266000
,HDRECSIZELNG     = 8 # 00267000

```

```

,FWCODESIZELOC      = 16 #      00268000
,FWCODESIZELNG      = 8 #      00269000
,NEWFWLEVELLOC      = 24 #      00270000
,NEWFWLEVELNG       = 4 #      00271000
,VENDORIDLOC        = 28 #      00272000
,VENDORIDLNG        = 8 #      00273000
,NUMDRIVEIDLOC      = 36 #      00274000
,NUMDRIVEIDLNG      = 4 #      00275000
,NUMREPFWLVLSLOC    = 40 #      00276000
,NUMREPFWLVLSLNG    = 4 #      00277000
,OLDFWLEVELNG       = 8 #      00278000
,DRIVEIDLNG         = 16 #      00279000
*,VENDORIDEXP        = "FUJITSU " #      00279500010040006
,VENDORIDEXP         = "UNISYS " #      00280000010040006
*,NOTESTRING         = "SBCFirmware." #      00281000
,TVENDORID          = ESHORTBUF [10] # % FOR 8      00282000
,TPRODID            = ESHORTBUF [18] # % FOR 16      00283000
,TPRODREVLVL        = ESHORTBUF [34] # % FOR 4      00284000
,TOEMID             = ESHORTBUF [54] # % FOR 8      00284500010020004
,TSERVOREVLVL       = ESHORTBUF [14] # % FOR 8      0028460001.005.000
;
;
; User Maint Requests
;
DEFINE
  ASSIGNUNITV        = 0 #      00285000
  ,UNASSIGNUNITV      = 1 #      00286000
  ,LOADSLAVEIMLV      = 2 #      00287000
  ,READERRORLOGV      = 3 #      00288000
  ,ATTRIBUTESV        = 4 #      00289000
  ,ASSIGNCTLV         = 5 #      00290000
  ,UNASSIGNCTLV       = 6 #      00291000
  ,MODESENSE          = 7 #      00292000
  ,DOWNLOADMODE4      = 8 #      00293000
  ,DOWNLOADMODE6      = 9 #      00294000
  ,DOWNLOADMODE7      = 10 #     00295000
  ,TESTUNITREADY      = 11 #     00296000
  ,INQUIRY            = 12 #     00296200010040006
;
;
;
LIBRARY
  MCP( FUNCTIONNAME    = "MCPSUPPORT.",
        LIBACCESS      = BYFUNCTION);
;
;
INTEGER PROCEDURE USERMAINTREQUEST(
  UNITNUMBER, FUNCTIONCODE, IOLENGTH, BUFFEROFFSET, BUFFERID,
  BUFFER, MRD, DEVICERD);
;
VALUE UNITNUMBER, FUNCTIONCODE, IOLENGTH, BUFFEROFFSET, BUFFERID;
;
INTEGER UNITNUMBER, FUNCTIONCODE, IOLENGTH, BUFFEROFFSET, BUFFERID, MRD;
;
EBCDIC ARRAY BUFFER[*], DEVICERD[*];
;
*****
; . Description:
;   This procedure is a Library-exported interface that supports
;   SCSI and IPI-3 I/O utilities (e.g., microcode download utility).
; . Parameters:
;   UNITNUMBER      External unit number of the device being
;   *               accessed. The MCP will determine whether a
;   *               port, CTL, or a device is being accessed using
;   *               data from the I/O unit tables.
;   FUNCTIONCODE:    Case index to be executed in the interface.
;   0: Assign Unit
;   1: Unassign Unit
;   2: Do FW Load (Load Slave IML)
;   3: Read Extended Status (Read Error Log)
;   4: Obtain Disk ID (Report Attributes)
;   *               5: Assign CTL
;   *               6: Unassign CTL
;   IOLENGTH        Bytes
;   BUFFEROFFSET     Used by SCSI interface to allow non-
;   *               transmittable data to precede actual data
;   *               in the buffer.
;
00300000
00301000
00302000
00303000
00304000
00305000
00306000
00307000
00308000
00309000
00310000
00311000
00312000
00313000
00314000
00315000
00316000
00317000
00318000
00319000
00320000
00321000
00322000
00323000
00324000
00325000
00326000
00327000
00328000
00329000

```

```

%*          Not required for IPI.                                00330000
%  BUFFERID      Not used by SCSI.                                00331000
%*          Not required for IPI.                                00332000
%  BUFFER        Direct EBCDIC array to be used as the source    00333000
%                of the data and the repository for extended     00334000
%                parameter handling.                              00335000
%  MRD           Maintenance Result Descriptor bits. Some bits   00336000
%                refer to HDPRESULT as returned to PTD.          00337000
%  DEVICERD      Status and Sense data                            00338000
%. Assumptions:                                              00339000
%  This routine protects the integrity of the system from a poorly 00340000
%  written application-level calling program.                  00341000
%. Results:                                              00342000
%  0:           Function succeeded.                               00343000
%  >0:          Function failed. See SHOWRSLT for details.      00344000
%. Locks:                                              00345000
%  None.                                                00346000
#####00347000
LIBRARY MCP;                                              00348000
                                              00349000
LABEL                                              00350000
MAINLOOP                                              00351000
,GRANDXIT;                                              00352000
$$ PAGE BEGINSEGMENT                                00353000
#####00353020010020004
%          CONVERT HEX TO DECIMAL                        00353040010020004
#####00353060010020004
PROCEDURE CONVERTHEXTODEC;                            00353080010020004
BEGIN                                              00353100010020004
                                              00353120010020004
INTEGER I;                                              00353140010020004
PTR      PTR;                                              00353160010020004
                                              00353180010020004
REAL                                              00353164010020004
TEMP                                              00353166010020004
,VAL;                                              00353168010020004
                                              00353180010020004
VAL := 0;                                              00353200010020004
PTR := SCR;                                              00353220010020004
                                              00353280010020004
FOR I := 1 STEP -1 UNTIL 0 DO                        00353300010020004
BEGIN                                              00353320010020004
IF PTR IN NUMBERS THEN                            00353340010020004
TEMP := INTEGER(PTR,1) * 16**I                    00353360010020004
ELSE                                              00353380010020004
CASE REAL(PTR,1) OF                                00353400010020004
BEGIN                                              00353420010020004
"A" : TEMP := 10 * 16**I;                          00353440010020004
"B" : TEMP := 11 * 16**I;                          00353460010020004
"C" : TEMP := 12 * 16**I;                          00353480010020004
"D" : TEMP := 13 * 16**I;                          00353500010020004
"E" : TEMP := 14 * 16**I;                          00353520010020004
"F" : TEMP := 15 * 16**I;                          00353540010020004
ELSE ; ;                                           00353560010020004
END;                                              00353580010020004
PTR := * + 1;                                       00353600010020004
VAL := * + TEMP;                                   00353620010020004
END;                                              00353640010020004
REPLACE OEM BY " " FOR 12 WORDS;                   00353660010020004
REPLACE OEM BY VAL FOR * DIGITS;                   00353680010020004
END; % PROCEDURE CONVERTHEXTODEC                    00353685010020004
#####00353690010020004
%          Common I/O Routines Block                    00353695010020004
#####00353700010020004
PROCEDURE SHOWIT(DSP);                                00353700010020004
VALUE      DSP;                                       00353700010020004
BOOLEAN    DSP;                                       00353700010020004
%-----00360000

```

```

% Purpose:
% Show L on LINE.
%
% Parameters:
% DSP      True: Caller wants a copy sent to user.
%
% Globals:
% ODTMODE  True: ODT is normal location of user.
%          False: Terminal is.
%
% Assumptions:
% L has message text.
% PL points to next place to add text.
% L will be blanked out upon exit.
-----
BEGIN
  DEFINE SEGMENT = #;

  IF NOT SKIPDTPRINT THEN
    WRITE(LINE,OFFSET(PL),L);      % Log the transaction
  IF DSP THEN                     % Send a copy to the user
    IF ODTMODE THEN              % at the console
      BEGIN
        REPLACE PL BY NUL;
        DISPLAY(L);
        WAIT((3));               % Slow down ODT traffic
      END
    ELSE                          % at his terminal
      WRITE(RMT,OFFSET(PL),L);
  BLANK;

END; % Show It

PROCEDURE PROMPT;
-----
% Purpose:
% Display text to the user and expect a response.
%
% Globals:
% ODTMODE  How we are to perform this action.
%
% Assumptions:
% L contains the output message.
% PL points to the next valid character in L.
% L needs to contain the input response.
% Both the output and input will be logged.
% The output will be upcased after logging in its original form.
% The output will be prescanned and deblanked using PL and LL.
% Under NO condition should PROMPT call SHOWIT. SHOWIT blanks out
% L as it exits, thus destroying all of our user input.
-----
BEGIN
  BOOLEAN BRSLT;

  IF ODTMODE THEN
    BEGIN
      WRITE(LINE,OFFSET(PL),L);      % Log it
      REPLACE PL BY NUL;
      ACCEPT(L);                     % Both output and input space
      SCAN L FOR LL:132 UNTIL = NUL;
      LL := 132 - LL;
      WRITE(LINE,LL,L);              % Log it again
    END
  ELSE % Terminal mode
    BEGIN
      WRITE(LINE,OFFSET(PL),L);      % Log it
      WRITE(RMT, OFFSET(PL),L);      % Tell user
      BRSLT := READ(RMT,80,L);
      IF BRSLT THEN                  % Error, maybe even ?END

```

```

00361000
00362000
00363000
00364000
00365000
00366000
00367000
00368000
00369000
00370000
00371000
00372000
00373000
00374000
00375000
00376000
00377000
00378000
00378500010010001
00379000010010001
00380000
00381000
00382000
00383000
00384000
00385000
00386000
00387000
00388000
00389000
00390000
00391000
00392000
00393000
00394000
00395000
00396000
00397000
00398000
00399000
00400000
00401000
00402000
00403000
00404000
00405000
00406000
00407000
00408000
00409000
00410000
00411000
00412000
00413000
00414000
00415000
00416000
00417000
00418000
00419000
00420000
00421000
00422000
00423000
00424000
00425000
00426000
00427000
00428000
00429000
00430000

```

```

GO GRANDXIT;
PL := L[LL:=REAL(BRSLT).[47:20]];
WRITE(LINE,LL,L);
END;

REPLACE L BY L FOR LL WITH UPCASER;
SCAN PL:L FOR LL:LL WHILE = " ";
END; % Prompt
$$ ENDSEGMENT
$$ PAGE BEGINSEGMENT
*****
% Debugging Module
*****
% Contents:
% SHOWMRDBITS
% SHOWHDPRESULT
% SHOWRSLT
% SHOWATTRIBUTES
*****
PROCEDURE SHOWMRDBITS;
% -----
% Purpose:
% Expand the Maintenance Result Descriptor bits following a call to
% USERMAINTREQUEST.
%
% Globals:
% MRD The Maintenance Result Descriptor
%
% Assumptions:
% The MRD is a packed bit field.
% We don't know how to decode this yet.
% This entire procedure will be replaced with defines, case statements,
% value arrays, etc.
% For the moment, just dump the MRD in hex.
*****
BEGIN
  DEFINE SEGMENT = #;

  REPLACE SCR BY MRD FOR 6;
  PUT(TXTLOC30)
  "MRD = ",
  HSCR FOR 12 WITH HEXTOEBCDIC;
  SHOWIT(TRUE);

END; % Show MRD bits

PROCEDURE SHOWHDPRESULT;
% -----
% Purpose:
% Expand the HDPRESULT coming out of the USERMAINTREQUEST.
%
% Globals:
% HDPRESULT A byte array containing RMM data.
%
% Assumptions:
% We think that only 12 bytes of data are valid. We may have
% gotten an interface error if our array is too short.
% We don't know the details of what is in this thing yet.
% For now, we will just format the first 12 bytes in hex.
% -----
BEGIN
  REPLACE SCR BY HDPRESULT FOR 12;
  PUT(TXTLOC30)
  "HDP = ",
  HSCR[00] FOR 12 WITH HEXTOEBCDIC,
  " ",
  HSCR[12] FOR 16 WITH HEXTOEBCDIC;
  SHOWIT(TRUE);

END; % Show HDP result

```

```

00431000
00432000
00433000
00434000
00435000
00436000
00437000
00438000
00439000
00440000
00441000
00442000
00443000
00444000
00445000
00446000
00447000
00448000
00449000
00450000
00451000
00452000
00453000
00454000
00455000
00456000
00457000
00458000
00459000
00460000
00461000
00462000
00463000
00464000
00465000
00466000
00467000
00468000
00469000
00470000
00471000
00472000
00473000010040006
00474000
00475000
00476000
00477000
00478000
00479000
00480000
00481000
00482000
00483000
00484000
00485000
00486000
00487000
00488000
00489000
00490000
00491000
00492000
00493000
00494000
00495000
00496000
00497000
0049800001.005.000
00499000010040006
00500000
00501000

```

PROCEDURE SHOWRSLT(RSLT,FCN);	00502000
VALUE RSLT,FCN;	00503000
INTEGER RSLT,FCN;	00504000
-----	00505000
% Purpose:	00506000
% Expand the integer value contained in RSLT from a USERMAINTREQUEST	00507000
% into understandable text.	00508000
%	00509000
% Parameters:	00510000
% RSLT The procedure value returned from USERMAINTREQUEST.	00511000
% FCN > 0 What function were we trying to do?	00512000
% FCN < 0 Elapsed time in Read Attributes.	00513000
%	00514000
% Assumptions:	00515000
% If RSLT = 0, the result was clean.	00516000
% If RSLT > 0, it is a CASE code for the actual error.	00517000
% We can use this to make global decisions in addition to	00518000
% just printing out the value.	00519000
-----	00520000
BEGIN	00521000
INTEGER MM,SS;	00522000
LABEL XIT;	00523000
	00524000
BLANK;	00525000
IF FCN < 0 THEN	% Often follows a PROMPT
BEGIN	% Format elapsed time
FCN := ABS(FCN);	00527000
MM := FCN DIV 60;	00528000
SS := FCN MOD 60;	00529000
START	00530000
MM FOR 2 DIGITS, ":",	00531000
%* SS FOR 2 DIGITS, " Read Attributes: ",	00532000
SS FOR 2 DIGITS, " Target: ",	00533000
CTLUNIT FOR * DIGITS,	00534000
" storing firmware.";	00535000
GO XIT;	00536000
END	00537000
ELSE	00538000
CASE FCN OF	00539000
BEGIN	00540000
ASSIGNUNITV:	00541000
START "Assign Unit: ", CTLUNIT FOR * DIGITS;	00542000
UNASSIGNUNITV:	00543000
START "Unassign Unit: ", CTLUNIT FOR * DIGITS;	00544000
LOADSLAVEIMLV:	00545000
START "Load Slave FW: ", CTLUNIT FOR * DIGITS;	00546000
READERRORLOGV:	00547000
START "Read Error Log: ", CTLUNIT FOR * DIGITS;	00548000
ATTRIBUTESV:	00549000
START "Read Attributes: ", CTLUNIT FOR * DIGITS;	00550000
ASSIGNCTLV:	00551000
START "Assign CTL: ", CTLUNIT FOR * DIGITS;	00552000
UNASSIGNCTLV:	00553000
START "Unassign CTL: ", CTLUNIT FOR * DIGITS;	00554000
END;	00555000
	00556000
PUT(TXTLOC22)	00557000
"RSLT = ",	00558000
RSLT FOR * DIGITS,	00559000
" : ";	00560000
CASE RSLT OF	00561000
BEGIN	00562000
0: ADD "No error";	00563000
1: ADD "Unit not supported (Vendor ID not UNISYS)";	00564000
3: ADD "Unit is not reserved (URed)";	00565000
5: ADD "Illegal/undefined function code specified";	00566000
7: ADD "I/O error. See MRD & DEVICERD parameters";	00567000
9: ADD "I/O length exceeds buffer's capacity";	00568000
11: ADD "UMR can't assign unit (reason unknown)";	00569000
13: ADD "Target is marked 'broken' by MCP";	00570000
	00571000
	00572000

15: ADD "Target is marked 'in use' by MCP";	00573000
17: ADD "Invalid target number was specified";	00574000
19: ADD "Unit not owned (FREE) by this partition";	00575000
21: ADD "I/O cannot be initiated";	00576000
23: ADD "UMR Can't determine target 'unit type'";	00577000
25: ADD "UMR issued I/O to unassigned target";	00578000
27: ADD "Unit not present (FREEd)";	00579000
29: ADD "Can't find usable path to target";	00580000
31: ADD "Illegal Unassign";	00580200010040006
33: ADD "IO length exceeds Max IO of 393216";	00580400010040006
ELSE ;	00581000
END;	00582000
XIT:	00583000
SHOWIT(TRUE);	00584000
END; % Show result	00585000
PROCEDURE SHOWATTRIBUTES;	00586000
-----	00587000
% Purpose:	00588000
% Expand the text in the SHORTBUF following a call to USERMAINTREQUEST	00589000
% with the ATTRIBUTESV parameter.	00590000
%	00591000
% Globals:	00592000
% SHORTBUF Contains the text results of ATTRIBUTES command.	00593000
%	00594000
-----	00595000
BEGIN	00596000
REPLACE SCR [0] BY " " FOR 2;	00597000
REPLACE POINTER(X[0]) BY SHORTBUF [54] FOR 1;	00598000
REPLACE SCR[0] BY POINTER(X[0],4) FOR 2 WITH HEXTOEBCDIC;	00599000
% REPLACE SCR BY "C8";	00600000
% SHOW (" SCR = " C SCR FOR 2, TRUE);	00600100010020004
CONVERTHEXTODEC;	00600200010020004
REPLACE ESHORTBUF [0] BY SHORTBUF [0] FOR SIZE (ESHORTBUF)	00600250010020004
WITH ASCIIIOEBCDIC;	00600300010020004
IF EXPRESSMODE THEN % Don't display - only print out	00600400010020004
BEGIN	00600500010020004
SHOW ("TARGET Attributes: " C	00600600010020004
"VENDOR ID - " C	00600700010020004
TVENDORID FOR VENDORIDLNG C ".", FALSE);	00601000
SHOW ("PRODUCT ID - " C TPRODID FOR DRIVEIDLNG C	00602000
"- " C OEM FOR 3 C ".", FALSE);	00602100010020003
SHOW ("FW LEVEL - " C	00602150010020003
TPRODREVLVL FOR NEWFWLEVELNG C ".", FALSE);	00602200010020003
END	00602250010020003
ELSE % Display and print	00602300010020003
BEGIN	00602350010020004
SHOW ("TARGET Attributes: " C	00602400010020004
"VENDOR ID - " C	00602450010020003
TVENDORID FOR VENDORIDLNG C ".", TRUE);	00602500010020003
SHOW ("PRODUCT ID - " C TPRODID FOR DRIVEIDLNG C	00602550010020003
"- " C OEM FOR 3 C ".", TRUE);	00602600010020003
SHOW ("FW LEVEL - " C	00602650010020003
TPRODREVLVL FOR NEWFWLEVELNG C ".", TRUE);	00603000010020003
END;	00604000010020003
END; % Show attributes	00605000010020003
PROCEDURE SHOWINQUIRYDATA;	00606000010020004
-----	00607000010020004
% Purpose:	00608000010020003
% Expand the text in the SHORTBUF following a call to USERMAINTREQUEST	00609000010020003
% with the INQUIRY parameter.	00610000
%	00611000
% Globals:	00612000
% SHORTBUF Contains the text results of ATTRIBUTES command.	0061202001.005.000
-----	0061204001.005.000
% Purpose:	0061206001.005.000
% Expand the text in the SHORTBUF following a call to USERMAINTREQUEST	0061208001.005.000
% with the INQUIRY parameter.	0061210001.005.000
%	0061212001.005.000
% Globals:	0061214001.005.000
% SHORTBUF Contains the text results of ATTRIBUTES command.	0061216001.005.000

[illegible]

```

BEGIN                                                    00626000
LABEL LOOPER;                                           00627000
                                                         00628000
                                                         00629000
                                                         00630000
**IF NOT ATODT THEN      % AT A TERMINAL
**BEGIN
SHOW ("DFAST is a firmware download utility for SCSI disk drives",TRUE);00631000
SHOW ("and/or SBC Controllers. DFAST requires:           ",TRUE);00632000
SHOW ("           ",TRUE);00633000
SHOW (" 1.) Firmware input file on disk or tape; e.g.,   ",TRUE);00634000
SHOW ("           ",TRUE);00635000
SHOW ("          (UCODE)FWFILE/123 ON ANYPACK - if on disk",TRUE);00636000
SHOW ("          or                                     ",TRUE);00637000
SHOW ("          (UCODE)FWFILE/123                      - if on tape",TRUE);00638000
SHOW ("           ",TRUE);00639000
SHOW (" 2.) A target (SCSI disk drive or SBC Controller) ",TRUE);00640000
SHOW ("          that has been reserved (URed).          ",TRUE);00641000
SHOW ("           ",TRUE);00642000
SHOW (" 3.) '<Mix #> AX' when responding to a prompt.    ",TRUE);00643000
SHOW ("           ",TRUE);00644000
IF NOT ATODT THEN
  WAIT ((5));                                           00645000
  SHOW ("The process follows the following format:       ",TRUE);00646000
  SHOW ("           ",TRUE);00647000
  SHOW ("           ",TRUE);00648000
  SHOW ("A.) The user is asked to select one of four options:",TRUE);00649000
  SHOW (" 1.) 'Ctlload' - select a SBC unit type.         ",TRUE);00650000010020003
  SHOW (" 2.) 'Devload' - select a SCSI disk drive unit type.",TRUE);0065100001.005.000
  SHOW (" 3.) 'Verifyfile' - check and display firmware file",TRUE);00652000010020003
  SHOW ("          header information without             ",TRUE);00653000010020003
  SHOW ("          performing a download to the target.",TRUE);00654000010020003
  SHOW (" 4.) 'Quit' - exit DFAST.                        ",TRUE);00655000010020003
  SHOW ("B.) The user is prompted for a firmware file name or",TRUE);00656000
  SHOW (" 'Quit' to exit DFAST.                          ",TRUE);00657000
  SHOW ("C.) If the file is validated as a firmware file, the new",TRUE);00658000
  SHOW (" firmware level is displayed; otherwise the program",TRUE);00659000
  SHOW (" returns to step 'B'.                             ",TRUE);00660000
IF NOT ATODT THEN
  WAIT ((5));                                           00661000
  SHOW ("D.) If 'Ctlload' or 'Diskload' was entered in step 'A'",TRUE);00662000
  SHOW (" then for every SCSI disk or SBC unit number entered,",TRUE);00663000
  SHOW (" steps 'a' through 'e' below are performed.      ",TRUE);00664000
  SHOW (" 'a.) Target attributes are obtained and displayed.",TRUE);00665000
  SHOW (" 'b.) If download is not allowed (due to mismatch of file/",TRUE);00666000
  SHOW (" target Firmware level, Product ID or Vendor ID), the",TRUE);00667000
  SHOW (" process stops for that unit, and the user is asked",TRUE);00668000
  SHOW (" to enter another target number of the same unit type",TRUE);00669000
  SHOW (" or 'End' to return to step 'A'.                   ",TRUE);00670000
  SHOW (" otherwise:                                         ",TRUE);00671000
  SHOW (" 'c.) If the SCSI drive (not SBC) requires formatting(IVR)",TRUE);00672000
  SHOW (" after the download, the user is consulted to:      ",TRUE);00673000010020004
  SHOW (" 1.) continue with the firmware download -or-       ",TRUE);00674000
  SHOW (" ii.) stop the process for that unit (to back up data",TRUE);00675000
  SHOW (" on that drive) and select another target of the    ",TRUE);00676000
  SHOW (" same unit type, or 'End' to return to step 'A'.     ",TRUE);00677000
  SHOW (" 'd.) After the download is complete and the target has",TRUE);00678000
  SHOW (" sequenced, the INQUIRY command is reissued, and the",TRUE);00679000
  SHOW (" new firmware level of the target is displayed.      ",TRUE);00680000
  SHOW (" 'e.) The user is prompted to enter another target number",TRUE);00681000
  SHOW (" of the same unit type or 'End' to return to step 'A'."",TRUE);00682000
IF NOT ATODT THEN
  WAIT ((5));                                           00683000
  **END
  $$ SET OMIT
  ELSE
    % AT AN A-SERIES CONSOLE
    BEGIN
    SHOW (" QUIT the program.                               ", TRUE); 00684000
    SHOW (" another device type or Firmware file or         ", TRUE); 00685000
    SHOW (" or SBC) or exit to outer block to select            ", TRUE); 00686000
    SHOW (" f.) Loop back for next unit of same type (disk       ", TRUE); 00687000
    SHOW (" e.) If error free, the process completes.            ", TRUE); 00688000
    SHOW (" the process for that unit.                             ", TRUE); 00689000
    SHOW (" either continue with downloading or stop             ", TRUE); 00690000

```

```

SHOW ("      after download, the user is consulted to ", TRUE); 00697000
SHOW (" d.) If the drive (not SBC) requires formatting ", TRUE); 00698000
SHOW ("      (IVR) otherwise ", TRUE); 00699000010020004
SHOW ("      Vendor ID) the process stops for that unit.", TRUE); 00700000
SHOW ("      of unit/file Firmware level, Product ID or ", TRUE); 00701000
SHOW (" c.) If download is not allowed (due to mismatch", TRUE); 00702000
SHOW (" b.) Unit information is obtained (INQUIRY). ", TRUE); 00703000
SHOW (" a.) The unit must be RESERVED (URed). ", TRUE); 00704000
SHOW (" for every Disk or SBC unit entered by the user: ", TRUE); 00705000
SHOW (" If the firmware file is read in properly, then ", TRUE); 00706000
SHOW ("The process follows the following format: ", TRUE); 00707000
SHOW (" ", TRUE); 00708000
SHOW (" 2.) SCSI Disk Drive or SBC Controller ", TRUE); 00709000
SHOW (" e.g., (UCODE)FWFILE/123 ON ANYPACK ", TRUE); 00710000
SHOW (" 1.) Firmware input file ", TRUE); 00711000
SHOW (" drives and/or SBC Controllers. DFAST requires: ", TRUE); 00712000
SHOW ("DFAST is a firmware download utility for SCSI disk", TRUE); 00713000
END; 00714000
$$ POP OMIT 00715000
00730000
READFCN; 00730200010020002
00730400010020002
00731000
LOOPER: 00732000
START " Do you want to continue? Enter 'YES' or 'NO' "; 00733000
PROMPT; 00734000
IF PL = "N" THEN 00735000
GO GRANDXIT 00736000
ELSE 00737000
IF PL NEQ "Y" THEN 00738000
GO LOOPER; 00739000
END; % BOOLEAN PROCEDURE USERPROCESS 00740000
00740200010020003
*****00740400010020003
00741000
BOOLEAN PROCEDURE OKTOFORMAT; 00742000
BEGIN 00743000
00744000
IF NOT EXPRESSMODE THEN 00744200010020003
BEGIN 00744400010020003
SHOW (" ", TRUE); 00745000
00746000
IF NOT ATODT THEN % AT A TERMINAL 00747000
BEGIN 00748000010020003
SHOW (">>>WARNING, DRIVE NEEDS FORMAT(IVR) AFTER DOWNLOAD", TRUE); 00749000010020004
SHOW (">>>WARNING, DRIVE NEEDS FORMAT(IVR) AFTER DOWNLOAD", TRUE); 00750000010020004
SHOW (">>>WARNING, DRIVE NEEDS FORMAT(IVR) AFTER DOWNLOAD", TRUE); 00751000010020004
SHOW (">>>WARNING, DRIVE NEEDS FORMAT(IVR) AFTER DOWNLOAD", TRUE); 00752000010020004
SHOW (">>>WARNING, DRIVE NEEDS FORMAT(IVR) AFTER DOWNLOAD", TRUE); 00753000010020004
END 00754000010020003
ELSE % AT AN A-SERIES CONSOLE 00755000
BEGIN 00756000010020003
SKIPODTPRINT := TRUE; % Display to console only - don't print 00756500010010001
SHOW (">>>WARNING, DRIVE NEEDS FORMAT(IVR) AFTER DOWNLOAD", TRUE); 00757000010020004
SHOW (">>>WARNING, DRIVE NEEDS FORMAT(IVR) AFTER DOWNLOAD", TRUE); 00758000010020004
SHOW (">>>WARNING, DRIVE NEEDS FORMAT(IVR) AFTER DOWNLOAD", TRUE); 00759000010020004
SHOW (">>>WARNING, DRIVE NEEDS FORMAT(IVR) AFTER DOWNLOAD", TRUE); 00760000010020004
SHOW (">>>WARNING, DRIVE NEEDS FORMAT(IVR) AFTER DOWNLOAD", TRUE); 00761000010020004
SKIPODTPRINT := FALSE; % Print only - don't display to console 00761100010010001
SHOW (">>>WARNING, DRIVE NEEDS FORMAT(IVR) AFTER DOWNLOAD", FALSE); 00761200010020004
SHOW (">>>WARNING, DRIVE NEEDS FORMAT(IVR) AFTER DOWNLOAD", FALSE); 00761300010020004
SHOW (">>>WARNING, DRIVE NEEDS FORMAT(IVR) AFTER DOWNLOAD", FALSE); 00761400010020004
SHOW (">>>WARNING, DRIVE NEEDS FORMAT(IVR) AFTER DOWNLOAD", FALSE); 00761500010020004
SHOW (">>>WARNING, DRIVE NEEDS FORMAT(IVR) AFTER DOWNLOAD", FALSE); 00761600010020004
END; 00762000010020003
00763000
SHOW (" ", TRUE); 00764000
SHOW (" USER BEWARE !!!! ", TRUE); 00765000
SHOW (" ", TRUE); 00766000
END; % IF NOT EXPRESSMODE 00766500010020003
00767000

```

SHOW ("Unit " C CTLUNIT FOR * DIGITS C " will need formatting " C	00770000010010001
"(IVR) after firmware is loaded!", TRUE);	00771000010020004
SHOW ("	00778000010010001
SHOW ("DO YOU STILL WANT TO CONTINUE WITH THE DOWNLOAD? ", TRUE);	00779000010010001
START "Answer 'YES' or 'NO'";	00780000010010001
PROMPT;	00781000
IF PL = "Y" THEN	00782000
OKTOFORMAT := TRUE	00783000
ELSE	00784000
OKTOFORMAT := FALSE;	00785000
END; % PROCEDURE OKTOFORMAT	00786000
BOOLEAN PROCEDURE VALIDATEACODE;	00787000
%-----	00788000
% Purpose:	00789000
% The user has received an unknown file from somewhere.	00790000
% He wants to see if it is a valid SBC firmware file.	00791000
%	00792000
% Procedure Result:	00793000
% True The code file is valid.	00794000
% False Invalid file for the target.	00795000
%	00796000
% Assumptions:	00797000
% We don't trust the caller.	00798000
%-----	00799000
BEGIN	00800000
REAL X;	00801000
INTEGER FK;	00802000
BOOLEAN RSLT;	00803000
LABEL XIT;	00804000
	00805000
RSLT := TRUE;	00806000
% Assume good file	00807000
SHOWIT(FALSE);	00808000
% Blank line	00809000
IF EXPRESSMODE THEN	00810000
SHOW("File Validation Report:", FALSE)	00810200010020003
ELSE	00810400010020003
SHOW("File Validation Report:", TRUE);	00810600010020003
	00811000010020003
START "Code file:";	00813000
PUT(TXTLOC15) CODE.TITLE;	00814000
IF EXPRESSMODE THEN	00815000
SHOWIT(FALSE)	00815200010020003
ELSE	00815400010020003
SHOWIT(TRUE);	00815600010020003
	00816000010020003
START "Customer ID:";	00817000
PUT(TXTLOC30)	00818000
">",	00819000
HDR[CUSTIDLOC] FOR CUSTIDLNG,	00820000
"<"	00821000
IF HDR[CUSTIDLOC] NEQ CUSTIDEXP THEN	00822000
BEGIN	00823000
SHOWIT(TRUE);	00824000
START " Expecting:";	00825000
PUT(TXTLOC30)	00826000
">",	00827000
CUSTIDEXP,	00828000
"<"	00829000
SHOWIT(TRUE);	00830000
RSLT := FALSE;	00831000
END	00832000
ELSE	00833000
IF EXPRESSMODE THEN	00834000
SHOWIT(FALSE)	00834200010020003
ELSE	00834400010020003
SHOWIT(TRUE);	00834600010020003
	00835000010020003
START "Vendor ID:";	00836000
PUT(TXTLOC30)	00837000
	00838000

```

">",
HDR[VENDORIDLOC] FOR VENDORIDLNG,
"<";
IF HDR[VENDORIDLOC] NEQ VENDORIDEXP THEN
BEGIN
  SHOWIT(TRUE);
  START " Expecting:";
  PUT(TXTLOC30)
  ">",
  VENDORIDEXP,
  "<";
  SHOWIT(TRUE);
  RSLT := FALSE;
END
ELSE
  IF EXPRESSMODE THEN
    SHOWIT(FALSE)
  ELSE
    SHOWIT(TRUE);

START "New Firmware Level:";
PUT(TXTLOC30)
">",
HDR[NEWFWLEVELLOC] FOR NEWFWLEVELNG,
"<";
IF EXPRESSMODE THEN
  SHOWIT(FALSE)
ELSE
  SHOWIT(TRUE);

XIT:
  VALIDATEACODE := RSLT;
  SHOWIT(FALSE);
END; % Validate Acode File

*****
% Compare Product IDs of Target and Input File
*****
BOOLEAN PROCEDURE COMPPRODIDS;
BEGIN
  BOOLEAN TARGETMATCH;
  INTEGER I, DRIVELOC;

  % Scan through valid Drive IDs in firmware file & match with Target

  I := 0;
  DRIVELOC := DRIVEIDLOC; % STARTING LOCATION IN HEADER RECORD

  WHILE ((I:= *+1) LEQ NUMDRIVEID) AND (NOT TARGETMATCH)) DO
    BEGIN
      TARGETMATCH := TPRODID = HDR [DRIVELOC] FOR DRIVEIDLNG;
      IF DEBUG THEN
        SHOW ("FILE PRODID-LOOP = " C HDR [DRIVELOC] FOR DRIVEIDLNG,TRUE);
      DRIVELOC := * + DRIVEIDLNG;
    END;

  IF NOT TARGETMATCH THEN
    BEGIN
      SHOW ("No match found in firmware file for Target " C
        CTLUNIT FOR * DIGITS C
        " with product ID = " C
        TPRODID FOR DRIVEIDLNG,TRUE);
      SHOW ("<< Download will NOT take place. >>",TRUE);
      COMPPRODIDS := FALSE;
    END
  ELSE
    COMPPRODIDS := TRUE;

END; % PROCEDURE COMPPRODIDS

```

```

00839000
00840000
00841000
00842000
00843000
00844000
00845000
00846000
00847000
00848000
00849000
00850000
00851000
00852000
00853000
00853200010020003
00853400010020003
00853600010020003
00854000010020003
00855000
00856000
00857000
00858000
00859000
00860000
00860200010020003
00860400010020003
00860600010020003
00861000010020003
00862000
00863000
00864000
00865000
00866000
00867000
00868000
00869000
00870000010010001
00871000
00872000
00873000010010001
00874000
00875000
00876000
00877000
00885000
00886000
00887000
00888000
00889000
00890000
00891000
00892000
00893000
00894000
00895000
00896000
00897000
00898000
00899000
00900000
00901000
00902000
00903000
00904000
00905000010010001
00906000
00907000
00908000010010001
00909000
00910000010010001

```

```

00911000
*****00912000
% Store Firmware File Information 00913000
*****00914000
00915000
PROCEDURE STOREFWFILEINFO; 00916000
BEGIN 00917000
00918000
INTEGER I; 00918200010040006
00918400010040006
% Assign Code file information from header record to program variables.00919000
00920000
REPLACE FCUSTID [0] BY HDR [CUSTIDLOC] FOR CUSTIDLNG; 00921000
REPLACE FNEWFWLEVEL [0] BY HDR [NEWFWLEVELLOC] FOR NEWFWLEVELNG; 00922000
REPLACE FVENDORID [0] BY HDR [VENDORIDLOC] FOR VENDORIDLNG; 00923000
FHDRBYTES := INTEGER (HDR [HDRECSIZELOC], HDRECSIZELNG); 00924000
FCODEBYTES := INTEGER (HDR [FWCODESIZELOC], FWCODESIZELNG); 00925000
NUMDRIVEID := INTEGER (HDR [NUMDRIVEIDLOC], NUMDRIVEIDLNG); 00926000
NUMREPFWLVLS := INTEGER (HDR [NUMREPFWLVLSLOC], NUMREPFWLVLSLNG); 00927000
DRIVEIDLOC := NUMREPFWLVLSLOC + NUMREPFWLVLSLNG + 00928000
(NUMREPFWLVLS * OLDFWLEVELNG); 00929000
00930000
IF DEBUG THEN 00931000
BEGIN 00932000
SHOW ("FROM FILE -- " C 00933000
"CUSTID : " C FCUSTID FOR CUSTIDLNG C 00934000
". NEW FW LEVEL : " C FNEWFWLEVEL FOR NEWFWLEVELNG,TRUE); 00935000
SHOW ("FROM FILE -- " C 00936000
"VENDORID : " C FVENDORID FOR VENDORIDLNG C 00937000
". HEADERBYTES : " C FHDRBYTES FOR HDRECSIZELNG DIGITS,TRUE); 00938000
SHOW ("FROM FILE -- " C 00939000
"CODEBYTES : " C FCODEBYTES FOR FWCODESIZELNG DIGITS C 00940000
". REPFWLVLS : " C NUMREPFWLVLS FOR NUMREPFWLVLSLNG DIGITS,TRUE); 00941000
SHOW ("FROM FILE -- " C 00942000
"NUMDRIVES : " C NUMDRIVEID FOR NUMDRIVEIDLNG DIGITS C 00943000
". DRIVEIDLOC: " C DRIVEIDLOC FOR DRIVEIDLNG DIGITS,TRUE); 00944000
END; 00945000
00945500010020004
IF NOT VERIFYONLY THEN 00946000010020004
00946200010020004
BEGIN 00946400010020004
% Resize the IML buffer 00947000
% SHOW("RESIZE THE IML BUFFER.",TRUE); 0094750001.005.000
IF FCODEBYTES >= SYSCAP THEN 0094800001.005.000
BEGIN 00948500010040006
NROWS := FCODEBYTES DIV EIGHTK; 00949000010040006
% REPLACE SCR[0] BY "NROWS= ", NROWS FOR * DIGITS; 0094902001.005.000
% WRITE(RMT,*/,SCR); 0094904001.005.000
IF NROWS >= MAXROWS THEN 0094906001.005.000
BEGIN 0094908001.005.000
RESIZE(IMLBUF2[*,*],MAXROWS,RETAIN); 0094920001.005.000
NROWS := MAXROWS; 0094925001.005.000
NROWS := * - 1; 0094930001.005.000
FOR I:= 0 STEP 1 UNTIL NROWS DO 0094940001.005.000
BEGIN 0094950001.005.000
SHOW(" I = " C I FOR * DIGITS,TRUE); 0094955001.005.000
RESIZE (IMLBUF2[I,*],EIGHTK,RETAIN); 0094960001.002.006
END; 0094980001.005.000
IMLP2 := IMLBUF2[0,*]; % ASSIGN POINTER 0095000001.005.001
NROWS := (FCODEBYTES - SYSCAP) DIV EIGHTK; 0095010001.005.001
NROWS := * + 1; 0095015001.005.000
RESIZE(IMLBUF3[*,*],NROWS,RETAIN); 0095020001.005.001
NROWS := * - 1; 0095030001.005.001
FOR I:= 0 STEP 1 UNTIL NROWS DO 0095040001.005.001
BEGIN 0095050001.005.001
SHOW(" I = " C I FOR * DIGITS,TRUE); 0095055001.005.000
RESIZE(IMLBUF3[I,*],EIGHTK,RETAIN); 0095060001.005.000
END; 0095070001.005.001
IMLP3 := IMLBUF3[0,*]; 0095080001.005.001
END 0095090001.005.001
END 00951000010040006

```

```

ELSE
    RESIZE(IMLBUF,FCODEBYTES,DISCARD);
    IMLP := IMLBUF;
%   SHOW("ASSIGN POINTER TO RESIZED IMLBUF.",TRUE);
%   Build IML image from Code File

IF NOT EXPRESSMODE THEN
    SHOW("Building FW image in buffer.",TRUE);

%   The LAST record of firmware code for SBC is transferred to the buffer
%   with the existing NULL padding at the end of that record. The SBC
%   parses out those NULLS in its internal buffer before processing the
%   data.

%   The LAST record of Firmware code for drives is transferred to the
%   buffer with only valid data (no NULLS are included). The drives
%   process the entire buffer as valid data.

IF FCODEBYTES >= 393216 THEN
BEGIN
    I := 0;
    SOFAR := 0;
    WHILE SOFAR < SYSCAP DO
    BEGIN
%   REPLACE SCR[0] BY "SOFAR= ", SOFAR FOR * DIGITS;
%   WRITE(RMT,*/,SCR);
%   DISPLAY (SCR);
        READ(CODE,CODEMRS,CODEREC);      % CODEMRS = 8192
        PC := CODEREC[0];                % ASSIGN POINTER PC TO CODEREC
        IF OPTODO = UNASSIGNCTLV THEN      % CODE IS FOR SBC
            REPLACE IMLBUF2[I,*] BY PC FOR CODEMRS
        ELSE
            REPLACE IMLBUF2[I,*] BY PC FOR CODEMRS
        BEGIN
            IF (FCODEBYTES - SOFAR) >= CODEMRS THEN % HAVE FULL DATA RECORD
                REPLACE IMLBUF2[I,*] BY PC FOR CODEMRS
            ELSE REPLACE IMLBUF2[I,*] BY PC
                FOR (FCODEBYTES - SOFAR);
            END;
            I := * + 1;
            SOFAR := * + CODEMRS;
        END;
    WHILE (FCODEBYTES - SOFAR) > 0 DO
    BEGIN
        I := 0;
        READ(CODE,CODEMRS,CODEREC);
        PC := CODEREC[0];
        IF OPTODO = UNASSIGNCTLV THEN
            REPLACE IMLBUF3[I,*] BY PC FOR CODEMRS
        ELSE
            BEGIN
                IF (FCODEBYTES - SOFAR) >= CODEMRS THEN
                    REPLACE IMLBUF3[I,*] BY PC FOR CODEMRS
                ELSE
                    REPLACE IMLBUF3[I,*] BY PC FOR (FCODEBYTES - SOFAR);
                END;
                I := * + 1;
                SOFAR := * + CODEMRS;
            END;
    END;
ELSE
BEGIN
    SOFAR := 0;
    WHILE SOFAR < FCODEBYTES DO
    BEGIN
        READ(CODE,CODEMRS,CODEREC);
        PC := CODEREC[0];
        IF OPTODO = UNASSIGNCTLV THEN      % CODE IS FOR SBC
            REPLACE IMLP:IMLP BY PC FOR CODEMRS
        ELSE
            REPLACE IMLP:IMLP BY PC FOR CODEMRS
        BEGIN
            IF (FCODEBYTES - SOFAR) >= CODEMRS THEN % HAVE FULL DATA RECORD

```

```

00951200010040006
00951400010040006
00951600010040006
0095180001.005.000
00952000
00953000
00953500010020003
00954000010020003
00955000
00956000
00957000
00958000
00959000
00960000
00961000
00962000
00963000
00963200010040006
0096340001.005.000
00963600010040006
00964500010040006
00965000010040006
0096600001.005.000
0096700001.005.000
0096720001.005.000
0096740001.005.000
0096760001.005.001
00968000010040006
00969000010040006
00970000
00971000010040006
00972000
00973000010040006
00974000010040006
00975000010040006
00976000010040006
00976500010040006
00977000010040006
00977500010040006
00978000
00979000
0097905001.005.000
0097910001.005.000
0097915001.005.000
0097920001.005.000
0097925001.005.000
0097930001.005.000
0097935001.005.000
0097940001.005.000
0097945001.005.000
0097950001.005.000
0097955001.005.000
0097960001.005.000
0097965001.005.000
0097970001.005.000
0097975001.005.000
0097980001.005.000
0097985001.005.000
00980000010040006
00980050010040006
00980100010040006
00980150010040006
00980200010040006
00980250010040006
00980300010040006
00980350010040006
00980400010040006
00980450010040006
00980500010040006
00980550010040006
00980600010040006

```

```

REPLACE IMLP:IMLP BY PC FOR CODEMRS                                00980650010040006
ELSE REPLACE IMLP:IMLP BY PC FOR (FCODEBYTES - SOFAR); % LAST    00980700010040006
END;                                                                % PARTIAL RECO0980750010040006
SOFAR := * + CODEMRS;                                             00980800010040006
END;                                                                00980850010040006
END;                                                                0098090001.002.004
END;                                                                00980950010040006
END; % PROCEDURE STOREFWFILEINFO                                  00981000
                                                                00982000
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%00983000
%          Procedure VERIFYFILE                                     %00984000
%          Procedure VERIFYFILE                                     %0098500001.005.000
                                                                00986000
PROCEDURE VERIFYFILE;                                             00987000
BEGIN                                                            00988000
  INTEGER N, X;                                                  00989000
  LABEL GETCODE;                                                 00990000
                                                                00991000
% Request name of code file                                       00992000
                                                                00993000
GETCODE:                                                         00994000
  CLOSE(CODE);                                                    00995000
  CODE.FILEUSE := VALUE(IN);                                       00996000
  CODE.KIND := VALUE(TAPE);                                        00997000
  CODE.DEPENDENTSPECS := TRUE;                                     00998000
  REPLACE TTL[0] BY " ";                                          00999000
  WHILE TTL[0] = " " DO                                          01000000
  BEGIN                                                            01001000
    START                                                         01002000
      "Enter name of code file or 'Quit'";                       01003000
    PROMPT;                                                       01004000
    IF LL > 0 THEN                                                01005000
    BEGIN                                                         01006000
      IF PL = "Q" THEN                                           01007000
      GO GRANDXIT;                                               01008000
      X := LL;                                                   01009000
      REPLACE L[LL+1] BY ".";                                     01010000
      SCAN PL FOR N:LL UNTIL = ".";                               01011000
      REPLACE TTL[0] BY PL FOR LL UNTIL = ".", ".";             01012000
      REPLACE CODE.TITLE BY TTL[0];                               01013000
      IF CODE.ATTERR THEN                                         01014000
      BEGIN                                                       01015000
        BLANK;                                                    01016000
        SHOW("Attribute error setting CODE file title",TRUE);   01017000
        GO GETCODE;                                               01018000
      END;                                                        01019000
    END; % LL > 0                                                 01020000
  END; % WHILE                                                    01021000
                                                                01022000
% Attempt to locate and open code file                            01023000
                                                                01024000
% A DISK file must have 'ON <PACKNAME>' in the title declaration 01025000
% (e.g., (UCODE)ABC/123 ON DISK), otherwise it defaults to a TAPE file. 01026000
% If the scan ends with a "." instead of a " ", then a tape type is 01027000
% assumed, because a disk file would require a " " for the "ON PACK" 01028000
% part of the title declaration.                                  01029000
                                                                01030000
SCAN PL:PL := TTL[0] FOR N:X UNTIL IN ENDER;                     01031000
                                                                01032000
IF PL = "." THEN % OPEN TAPE                                     01033000
  OPEN (CODE) % ALLOW FOR TAPE FILE EQUATE                       01034000
ELSE IF CODE.AVAILABLE = 1 THEN % DISK FILE                     01035000
  CODEEOF := CODE.LASTRECORD                                     01036000
ELSE % DISK FILE NOT AVAILABLE                                   01037000
  BEGIN                                                         01038000
    SHOW ("The disk file name entered is NOT AVAILABLE", TRUE); 01039000
    GO GETCODE;                                                  01040000
  END;                                                           01041000
                                                                01042000
CODEMRS := EIGHTK;                                               01043000
IF SIZE(HDR) < CODEMRS THEN                                       01044000

```

```

BEGIN
RESIZE(HDR,CODEMRS,DISCARD);
RESIZE(CODEREC,CODEMRS,DISCARD);
END;

READ(CODE,CODEMRS,CODEREC);
REPLACE HDR BY CODEREC FOR SIZE(HDR) WITH ASCII TO EBCDIC;

% See if code file is a valid TARGET code file

IF VALIDATECODE THEN
BEGIN
IF EXPRESSMODE THEN
SHOW("File is a valid TARGET code file",FALSE)
ELSE
SHOW("File is a valid TARGET code file",TRUE);
SHOWIT(FALSE);
STOREFWFILEINFO;
END
ELSE
BEGIN
SHOW("File is -not- a valid TARGET code file",TRUE);
SHOWIT(FALSE);
GO GETCODE;
END;

END; % Verify File

*****
% Compare Firmware Levels of Target and Input File
*****

BOOLEAN PROCEDURE COMPFWLEVELS;
BEGIN

INTEGER FWFILELOC, I;
BOOLEAN FWLEVELSMATCH;

I := 0;
% WANT LOWER FOUR BYTES (3,2,1,0) OF 8 BYTE FIELD

FWFILELOC := NUMREPFWLVL; % OFFSET BY '4' FOR PROPER LOOPING
% INCREMENT BELOW.
WHILE ((I:=I+1) LEQ NUMREPFWLVL) AND (NOT FWLEVELSMATCH) DO
BEGIN
FWLEVELSMATCH := TPRODREVLVL = HDR [(FWFILELOC:=I+1) * OLDFWLEVELNG]
FOR NEWFWLEVELNG;
IF DEBUG THEN
% SHOW ENTIRE 7 OR 8 BYTES OF FILE ENTRY
% IF NULLS ARE PRINTED, OUTPUT IS LOST.
IF OPTODO = UNASSIGNUNITY THEN
BEGIN
IF HDR [(FWFILELOC - 4)] = 48"00" THEN % BYTE [0] = NULL
SHOW ("FILE FW-LOOP = >" C HDR [(FWFILELOC - 3)]
FOR (OLDFWLEVELNG - 1) C "<", TRUE)
ELSE
% BYTE [0] NEQ NULL. FORMAT REQUIRED FOR SCSI DRIVES
SHOW ("FILE FW-LOOP = >" C HDR [(FWFILELOC - 4)]
FOR OLDFWLEVELNG C "<", TRUE);
END
ELSE
% FOR SBC
SHOW ("FILE FW-LOOP = >" C HDR [FWFILELOC] FOR NEWFWLEVELNG
C "<", TRUE);
END;

IF NOT FWLEVELSMATCH THEN % NO MATCH FOUND
BEGIN
SHOW ("<< Firmware level of target is NOT in input file. >>",TRUE);
SHOW ("<< Download will NOT take place. >>", TRUE);
COMPFWLEVELS := FALSE;
END
ELSE
BEGIN

```

```

01045000
01046000
01047000
01048000
01049000
01050000
01051000
01052000
01053000
01054000
01055000
01056000
01056200010020003
01056400010020003
01056600010020003
01057000010020003
01058000
01059000
01060000
01061000
01062000
01063000
01064000
01065000
01066000
01067000
01068000
01069000
01070000
01071000
01072000
01073000
01074000
01075000
01076000
01077000
01078000
01079000
01080000
01081000
01082000
01083000
01084000
01085000
01086000
01087000
01088000
01089000
01089100010010001
01089200010010001
01089400010010001
01090000
01091000
01092000
01093000
01094000
01095000
01095200010010001
01095400010010001
01095600010010001
01095800010010001
01096000
01097000
01098000
01099000
01100000
01101000
01102000
01103000
01104000
01105000

```

```

IF OPTODO = UNASSIGNUNITV THEN      % Drives only. Get full 8 bytes
BEGIN
REPLACE SAVEFWLVL [0] BY HDR [(FWFILELOC - 4)] FOR OLDFWLEVELNG;
IF DEBUG THEN
SHOW ("SAVEFWLVL (FORMAT(IVR) NEEDED?-BYTE[0] NEQ 48'00') = >"
SAVEFWLVL FOR OLDFWLEVELNG C "<", TRUE);
END;
COMPFWLEVELS := TRUE;
END;

END; % PROCEDURE COMPFWLEVELS
*****
% Compare SERVO Firmware Levels of Target and Input File
*****
BOOLEAN PROCEDURE COMPSERVOFWLVL;
BEGIN

INTEGER FWFILELOC, I;
BOOLEAN FWLEVELSMATCH;

I := 0;
% WANT LOWER FOUR BYTES (3,2,1,0) OF 8 BYTE FIELD

FWFILELOC := NUMREPFWLVL; % OFFSET BY '4' FOR PROPER LOOPING
% INCREMENT BELOW.
WHILE ((I:= *+1) LEQ NUMREPFWLVL) AND (NOT FWLEVELSMATCH) DO
BEGIN
FWLEVELSMATCH := TSERVOREVLVL = HDR [(FWFILELOC:= *+ OLDFWLEVELNG)]
FOR NEWFWLEVELNG;
% SHOW("SERVO FIRMWARE =>" C
% TSERVOREVLVL FOR NEWFWLEVELNG, TRUE);
IF DEBUG THEN % SHOW ENTIRE 7 OR 8 BYTES OF FILE ENTRY
% IF NULLS ARE PRINTED, OUTPUT IS LOST.
IF OPTODO = UNASSIGNUNITV THEN
BEGIN
IF HDR [(FWFILELOC - 4)] = 48"00" THEN % BYTE [0] = NULL
SHOW ("FILE FW-LOOP = >" C HDR [(FWFILELOC - 3)]
FOR (OLDFWLEVELNG - 1) C "<", TRUE)
ELSE % BYTE [0] NEQ NULL. FORMAT REQUIRED FOR SCSI DRIVES
SHOW ("FILE FW-LOOP1 = >" C HDR [(FWFILELOC - 4)]
FOR OLDFWLEVELNG C "<", TRUE);
END
ELSE % FOR SBC
SHOW ("FILE FW-LOOP2 = >" C HDR [FWFILELOC] FOR NEWFWLEVELNG
C "<", TRUE);
END;

COMPSERVOFWLVL := TRUE;

END; % PROCEDURE COMPFWLEVELS
$$ PAGE

*****
% Release the target
*****
BOOLEAN PROCEDURE RELEASETARGET (OPTODO);
VALUE OPTODO;
INTEGER OPTODO;
BEGIN

INTEGER N, RSLT;

% Release the target

RSLT := USERMAINTREQUEST(CTLUNIT,OPTODO,0,0,0,
SHORTBUF,MRD,HDPRESULT);
IF RSLT > 0 THEN
BEGIN
SHOW("Unable to release TARGET " C

```

CTLUNIT FOR * DIGITS	C	01137000
" Error "	C	01138000
RSLT FOR * DIGITS,TRUE);		01139000
RELEASETARGET := FALSE;		01140000
END		01141000
ELSE		01142000
BEGIN		01143000
RELEASETARGET := TRUE;		01144000
IF OPTODO = UNASSIGNUNITV THEN		01145000
BEGIN		01146000
SDMASSIGNED := FALSE;		01147000
SHOWRSLT (RSLT, UNASSIGNUNITV);		01148000
END		01149000
ELSE		01150000
BEGIN		01151000
CTLASSIGNED := FALSE;		01152000
SHOWRSLT (RSLT, UNASSIGNCTLV);		01153000
END;		01154000
END;		01155000
		01156000
END; % BOOLEAN PROCEDURE RELEASETARGET		01157000
*****		01158000
% Load SBC Firmware Procedure		01159000
*****		01160000
		01161000
PROCEDURE LOADCTLFW;		01162000
BEGIN		01163000
INTEGER N,RSLT,W,OFFSET,I;		01164000010040006
LABEL NEXTCTL;		01165000
ARRAY T[0:2];		01166000010030005
		01166400010030005
% System call to determine machine type		01167000
% If we are not running on an IOM type system, terminate.		01168000
		01169000
SYSTEMTYPE := TIME(24);		01170000010030005
REPLACE T BY SYSTEMTYPE FOR 6 ;		01170500010040007
IF T ISNT "A11" FOR 3 AND		01171000010030005
T ISNT "A14" FOR 3 AND		01171500010030005
T ISNT "A12" FOR 3 AND		01171600010040006
T ISNT "A16" FOR 3 AND		01172000010040007
T ISNT "A18" FOR 3 AND		01172500010030005
T ISNT "A19" FOR 3 AND		01173000010030005
T ISNT "A28" FOR 3 AND		01173500010040007
T ISNT "NX46" FOR 4 AND		01173600010040007
T ISNT "NX48" FOR 4 AND		011738001.005.000
T ISNT "NX58" FOR 4. THEN		011739001.005.000
BEGIN		01174000
SHOW("SYSTEMTYPE = " C SYSTEMTYPE FOR 6, TRUE);		01174500010030005
SHOW("Running system is not an IOM system. Program is ENDING",TRUE);		01175000010010001
GO GRANDXIT;		01176000
END;		01177000
SHOW("SYSTEMTYPE = " C SYSTEMTYPE FOR 6, TRUE);		01178000010030005
START		01179000
"Caution: download FW to only one CTL in a string at a time";		01180000
SHOWIT(TRUE);		01181000
START " CTL must be URed to all visible hosts";		01182000
SHOWIT(TRUE);		01183000
START " Other CTL must be online";		01184000
SHOWIT(TRUE);		01185000
		01186000
***** MAIN TARGET (CTL) LOOP *****		01187000
		01188000
OPTODO := UNASSIGNCTLV; % CTL PARAMETER TO UMR		01189000
		01190000
WHILE TRUE DO		01191000
BEGIN		01192000
%Request CTL number		01193000
		01194000
CTLUNIT := 0;		01195000
WHILE CTLUNIT = 0 DO		01196000
BEGIN		01197000

[illegible]

SHOWATTRIBUTES;	01241000
% Get code file & match header rec. info against Target info.	01242000
	01243000
IF TVENDORID NEQ VENDORIDEXP FOR VENDORIDLNG THEN	01244000
BEGIN	01245000
SHOW ("Vendor ID of target is not UNISYS.", TRUE);	01246000
SHOW ("<< Download will NOT take place. >>", TRUE);	01247000
IF NOT RELEASETARGET (OPTODO) THEN	01248000
GO GRANDXIT	01249000
ELSE	01250000
GO NEXTCTL;	01251000
END;	01252000
IF NOT COMPPRODIDS THEN % COMPARE PRODUCT IDs OF FILE VS. TARGET	01253000
IF NOT RELEASETARGET (OPTODO) THEN	01254000
GO GRANDXIT	01256000010010001
ELSE	01257000
GO NEXTCTL;	01258000
	01259000
	01260000
	01261000
IF NOT COMPFWLEVELS THEN % COMPARE FIRMWARE LEVELS OF FILE VS. TARGET	01262000
IF NOT RELEASETARGET (OPTODO) THEN	01263000
GO GRANDXIT	01264000
ELSE	01265000
GO NEXTCTL;	01266000
	01267000
IF TPRODREVLVL = FNEWFWLEVEL FOR NEWFWLEVELNG THEN	01268000
BEGIN	01269000
SHOW ("Firmware levels of Target and File are the same.", TRUE);	01270000
START "Do you still want to download the firmware? Enter YES or NO";	01271000
PROMPT;	01272000
IF PL NEQ "Y" THEN	01273000
BEGIN	01274000
SHOW ("Download will not take place for target " C	01275000
CTLUNIT FOR * DIGITS, TRUE);	01276000
IF NOT RELEASETARGET (OPTODO) THEN	01277000
GO GRANDXIT	01278000
ELSE	01279000
GO NEXTCTL;	01280000
END;	01281000
END;	01282000
% Request function	01283000
	01284000
	01285000
SHOW("Starting to download code to CTL " C	01286000
CTLUNIT FOR * DIGITS, TRUE);	01287000
IF SYSCAP <= FCODEBYTES THEN	01287050010040006
BEGIN	01287100010040006
NUMBROFIOS := FCODEBYTES DIV EIGHTK;	01287150010040006
SIZEOFLSTIO := FCODEBYTES MOD EIGHTK;	01287200010040006
OFFSET := 0;	01287250010040006
I := 0;	01287300010040006
DO	01287350010040006
BEGIN	01287400010040006
RSLT := USERMAINTREQUEST(CTLUNIT, DOWNLOADMODE6, 8192, OFFSET, 0,	01287450010040006
IMLBUF2[I,*], MRD, HDPRESULT);	01287500010040006
IF RSLT > 0 THEN	01287550010040006
BEGIN	01287600010040006
SHOWRSLT(RSLT, LOADSLAVEIMLV);	01287650010040006
SHOWMRDBITS;	01287700010040006
SHOWHDPRESULT;	01287750010040006
	01287800010040006
SHOW("<< Microcode NOT loaded!! >>", TRUE);	01287850010040006
IF NOT RELEASETARGET (OPTODO) THEN	01287900010040006
GO GRANDXIT;	01287950010040006
END;	01288000010040006
REPLACE SCR[0] BY "I= ", I FOR *DIGITS;	01288010010040006
DISPLAY (SCR);	01288020010040006
REPLACE SCR2[0] BY "NUMBROFIOS= ", NUMBROFIOS FOR * DIGITS;	01288030010040006
DISPLAY (SCR2);	01288040010040006
NUMBROFIOS := * - 1;	01288050010040006

```

        OFFSET := * + EIGHTK;
        I := * + 1;
    END
    UNTIL NUMBROFIOS = 1;
    RSLT := USERMAINTREQUEST(CTLUNIT,DOWNLOADMODE7,8192,OFFSET,0,
        IMLBUF2[I,*],MRD,HDPRESULT);
END
ELSE
    RSLT := USERMAINTREQUEST(CTLUNIT,LOADSLAVEIMLV,FCODEBYTES,0,0,
        IMLBUF,MRD,HDPRESULT);
IF RSLT > 0 THEN
    BEGIN
        SHOWRSLT(RSLT,LOADSLAVEIMLV);
        SHOWMRDBITS;
        SHOWHDPRESULT;

        SHOW("<< Microcode NOT loaded!! >>",TRUE);
        IF NOT RELEASETARGET (OPTODO) THEN
            GO GRANDXIT
        ELSE
            GO NEXTCTL;
        END;

        SHOW("Download complete. Waiting to read attributes of CTL " C
            CTLUNIT FOR * DIGITS,TRUE);
        SHOW("Do not power off or alter CTL " C
            CTLUNIT FOR * DIGITS,TRUE);

% The SBC has now turned off its SCSI interface.
% It won't come alive until after it has done the power up
% confidence tests (approx. 30 seconds).
% Initially wait 30 seconds and then cycle ATTRIBUTES calls
% every 10 seconds until it comes back to life or drops dead.

        WHEN(30);
        SHOW("00:30 Starting to read attributes of CTL " C
            CTLUNIT FOR * DIGITS,TRUE);
        TIMER := 0;
        REPLACE SHORTBUF [0] BY NUL FOR SIZE (SHORTBUF);

        RSLT := USERMAINTREQUEST(CTLUNIT,TESTUNITREADY,0,0,0,
            SHORTBUF,MRD,HDPRESULT);

        IF RSLT > 0 THEN
            BEGIN
                SHOWRSLT(RSLT,LOADSLAVEIMLV);
                SHOWMRDBITS;
                SHOWHDPRESULT;

                SHOW("<< Test Unit Ready Failed!! >>",TRUE);
                IF NOT RELEASETARGET (OPTODO) THEN
                    GO GRANDXIT;
                END;
            END;
        DISPLAY(" Test Unit Ready - OK ");

        DO BEGIN
            WHEN(10);
            RSLT := USERMAINTREQUEST(CTLUNIT,ATTRIBUTESV,254,0,0,
                SHORTBUF,MRD,HDPRESULT);
            N := (TIMER:=*+1)*10 + 30;           % Seconds
            SHOWRSLT(RSLT,-N);

        END UNTIL N >= 1*60           % Drop dead time
            OR RSLT = 0;           % Success

        IF RSLT > 0 THEN           % Timeout
            BEGIN
                SHOW("MCP interface error " C
                    RSLT FOR * DIGITS           C
                    " after FW download"         C
                    " timeout on CTL "           C

```

```

01288100010040006
01288150010040006
01288300010040006
01288350010040006
01288400010040006
01288450010040006
01288500010040006
01288550010040006
01288600010040006
01288650010040006
01290000
01291000
01292000
01293000
01294000
01295000
01296000
01297000
01298000
01299000
01300000
01301000
01302000
01303000
01304000
01305000
01306000
01307000
01308000
01309000
01310000
01311000
01312000
01313000
01314000
01315000
01316000
01317000
01318000
01318050010040006
01318100010040006
01318150010040006
01318200010040006
01318250010040006
01318300010040006
01318350010040006
01318400010040006
01318450010040006
01318500010040006
01318550010040006
01318600010040006
01318650010040006
01318700010040006
01318750010040006
01319000
01320000
01321000
01322000
01323000
01324000
01325000
01326000
01327000
01328000
01329000
01330000
01331000
01332000
01333000
01334000
01335000

```

```

        CTLUNIT FOR * DIGITS,TRUE);
SHOWRSLT(RSLT,ATTRIBUTESV);
SHOW("Check CTL for malfunction",TRUE);
GO GRANDXIT;
END;

SHOW("CTL "          C
      CTLUNIT FOR * DIGITS  C
      " completed FW download",TRUE);

% Format attributes

SHOWATTRIBUTES;

SHOW ("          ", TRUE);
SHOW ("If BOOT code was loaded instead of OPERATIONAL code", TRUE);
SHOW ("The firmware level of the SBC may not have changed ", TRUE);
SHOW ("          ", TRUE);

% Release the CTL

IF NOT RELEASETARGET (OPTODO) THEN
GO GRANDXIT;

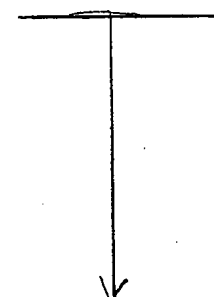
SHOW("Okay to UR- CTL "      C
      CTLUNIT FOR * DIGITS  C
      " and load companion CTL",TRUE);

NEXTCTL:
END; % WHILE TRUE DO

END; % Load CTL Firmware

#####01336000
%          Load SCSI Drive Firmware Procedure          01337000
#####01338000
PROCEDURE LOADEVFW;          01339000
BEGIN          01340000
    INTEGER N,RSLT,W,OFFSET,I,J;          01341000
    LABEL NEXTDRIVE;          01342000
    ARRAY T(0:2);          01343000
% System call to determine machine type          01344000
% If we are not running on an IOM type system, terminate.          01348000
          01349000
          01350000
SYSTEMTYPE := TIME(24);          01351000
% REPLACE T BY SYSTEMTYPE FOR 3 ;          01351100010010001
% IF T ISNT "A11" FOR 3 AND          01351200010010001
%   T ISNT "A14" FOR 3 AND          01351300010010001
%   T ISNT "A16" FOR 3 AND          01351400010010001
%   T ISNT "A17" FOR 3 AND          01351500010010001
%   T ISNT "A18" FOR 3 AND          01352000
%   T ISNT "A19" FOR 3 AND          01353000
%   T ISNT "A28" FOR 3 THEN          01354000
% BEGIN          01355000
%   SHOW("SYSTEMTYPE = " C SYSTEMTYPE FOR 6, TRUE);          01356000
%   SHOW("Running system does not support this drives." C          01357000
%     " Program is ENDING.",TRUE);          01357200010020002
%   GO GRANDXIT;          01357400010020002
% END;          01357600010020002
SHOW("SYSTEMTYPE = " C SYSTEMTYPE FOR 6,TRUE);          01357800010020002
IF NOT EXPRESSMODE THEN          01358000
    BEGIN          01359000
        START "Caution: Load only one drive in a string at a time";          01360000
        SHOWIT(TRUE);          01361000
        START "          drive must be URed to all visible hosts";          01362000
        SHOWIT(TRUE);          01363000
        START "          If using Fibre Channel UR the opposite Port";          01364000
        SHOWIT(TRUE);          01365000001.005.001
        END;          01366000
    END;          01367000
END;          01368000001.005.001
          01369000
          01370000001.005.001
          01371000
          013720000010030005
          01373000
          01374000
          01375000
          013760000010030005
          013765000010030005
          013770000010030005
          013775000010030005
          013780000010030005
          013790000010030005
          013795000010030005
          013800000010030005
          013805000010030005
          013810000010030005
          013815000010030005
          013820000010030005
          013830000010030005
          013840000010030005
          013850000010030005
          013860000010030005
          013862000010020003
          013864000010020003
          013870000010020003
          013880000010020003
          013890000010020003
          013900000010020003
          0139020001.005.000
          0139040001.005.000
          01390500010020003

```



[illegible]

IF RSLT > 0 THEN	% Problem	01433000
BEGIN		01434000
SHOW("MCP interface error "	C	01435000
RSLT FOR * DIGITS	C	01436000
" to get target"	C	01437000
" attributes for SCSI drive " C		01438000
CTLUNIT FOR * DIGITS,TRUE);		01439000
SHOWRSLT(RSLT,ATTRIBUTESV);		01440000
SHOW("Check TARGET for problem",TRUE);		01441000
IF NOT RELEASETARGET (OPTODO) THEN		01442000
GO GRANDXIT		01443000
ELSE		01444000
GO NEXTDRIVE;		01445000
END;		01446000
SHOWATTRIBUTES;		01447000
		01448000
		01449000
IF EXPRESSMODE THEN	% Save for display at end of download	01449200010020003
REPLACE OLDFWLEVEL [0] BY TPRODREVLVL FOR NEWFWLEVELNG;		01449400010020003
		01449600010020003
% Get code file & match header rec. info against Target info.		01450000
		01451000
IF TVENDORID NEQ VENDORIDEXP FOR VENDORIDLNG THEN		01452000
BEGIN		01453000
SHOW ("Vendor ID of target is not UNISYS.", TRUE);		01454000
SHOW ("<< Download will NOT take place. >>", TRUE);		01455000
IF NOT RELEASETARGET (OPTODO) THEN		01456000
GO GRANDXIT		01457000
ELSE		01458000
GO NEXTDRIVE;		01459000
END;		01460000
IF NOT COMPPRODIDS THEN	% COMPARE PRODUCT IDs OF FILE vs. TARGET	01462000010010001
IF NOT RELEASETARGET (OPTODO) THEN		01463000
GO GRANDXIT		01464000
ELSE		01465000
GO NEXTDRIVE;		01466000
		01467000
IF TPRODREVLVL = FNEWFWLEVEL FOR NEWFWLEVELNG THEN		0147400001.005.000
BEGIN		01475000
SHOW ("Firmware levels of Target and File are the same.",TRUE);		01476000
START "Do you still want to download the firmware? Enter YES or NO";		01477000
PROMPT;		01478000
IF PL NEQ "Y" THEN		01479000
BEGIN		01480000
SHOW ("Download will not take place for target " C		01481000
CTLUNIT FOR * DIGITS, TRUE);		01482000
IF NOT RELEASETARGET (OPTODO) THEN		01483000
GO GRANDXIT		01484000
ELSE		01485000
GO NEXTDRIVE;		01486000
END		01487000
ELSE REPLACE SAVEFWLVL [0] BY 48"00" FOR OLDFWLEVELNG;	% NO FORMAT	01487500010010001
END		01488000
ELSE		01489000
IF NOT COMPEFWLEVELS THEN	% COMPARE FW LEVELS OF FILE VS. TARGET	0148920001.005.000
IF NOT RELEASETARGET (OPTODO) THEN		01489400010010001
GO GRANDXIT		01489600010010001
ELSE		01489800010010001
GO NEXTDRIVE;		01490000010010001
		01490200010010001
% If drive needs to be formatted after firmware download, get		01491000010010001
% permission before downloading the code.		01492000010010001
% Each FW level area in file = 8 bytes. If byte 0 neq 48"00", drive		01493000010010001
% must be formatted after code is loaded.		01494000010010001
IF SAVEFWLVL [0] NEQ 48"00" THEN	% Drive needs formatting	01495000010010001
IF NOT OKTOFORMAT THEN		01496000010010001
IF NOT RELEASETARGET (OPTODO) THEN		01497000010010001
GO GRANDXIT		01498000
ELSE		01499000
GO NEXTDRIVE;		01500000
		01502000

```

% Get attributes of Target
RSLT := USERMAINTREQUEST(CTLUNIT,INQUIRY,23,192,0,
                          SHORTBUF,MRD,HDPRESULT);
IF RSLT > 0 THEN
  BEGIN
    SHOW("MCP interface error " C
          RSLT FOR * DIGITS C
          " to get target" C
          " attributes for SCSI drive " C
          CTLUNIT FOR * DIGITS,TRUE);
    SHOWRSLT(RSLT,ATTRIBUTESV);
    SHOW("Check TARGET for problem",TRUE);
    IF NOT RELEASETARGET (OPTODO) THEN
      GO GRANDXIT
    ELSE
      GO NEXTDRIVE;
  END;

  SHOWINQUIRYDATA;

  IF EXPRESSMODE THEN
    REPLACE OLDFWLVL [0] BY TSERVOREVLVL FOR NEWFWLEVELNG;

% Get code file & match header rec. info against Target info.

IF TSERVOREVLVL = FNEWFWLEVEL FOR NEWFWLEVELNG THEN
  BEGIN
    SHOW ("Servo FW levels of Target and File are the same.",TRUE);
    START "Do you still want to download the firmware? Enter YES or NO";
    PROMPT;
    IF PL NEQ "Y" THEN
      BEGIN
        SHOW ("Download will not take place for target " C
              CTLUNIT FOR * DIGITS, TRUE);
        IF NOT RELEASETARGET (OPTODO) THEN
          GO GRANDXIT
        ELSE
          GO NEXTDRIVE;
      END
    ELSE REPLACE SAVEFWLVL [0] BY 48"00" FOR OLDFWLVLNG; % NO FORMAT
  END
ELSE
  IF NOT COMPSERVOFWLVL THEN % COMPARE FW LEVELS OF FILE VS. TARGET
    IF NOT RELEASETARGET (OPTODO) THEN
      GO GRANDXIT
    ELSE
      GO NEXTDRIVE;

% If drive needs to be formatted after firmware download, get
% permission before downloading the code.
% Each FW level area in file = 8 bytes. If byte 0 neq 48"00", drive
% must be formatted after code is loaded.
IF SAVEFWLVL [0] NEQ 48"00" THEN % Drive needs formatting
  IF NOT OKTOFORMAT THEN
    IF NOT RELEASETARGET (OPTODO) THEN
      GO GRANDXIT
    ELSE
      GO NEXTDRIVE;

% Request function

IF NOT EXPRESSMODE THEN
  SHOW("Starting to download code to drive " C
        CTLUNIT FOR * DIGITS,TRUE);
IF FCODEBYTES > SYSCAP THEN
  BEGIN
    NUMBROFIOS := FCODEBYTES DIV EIGHTK;
    NUMBROFIOS := * + 1;
    SHOW(" NUMBROFIOS = " C NUMBROFIOS FOR * DIGITS,TRUE);

```

```

0150201001.005.001
0150202001.005.001
0150203001.005.000
0150204001.005.001
0150205001.005.001
0150206001.005.001
0150207001.005.001
0150208001.005.001
0150209001.005.001
0150210001.005.001
0150211001.005.001
0150212001.005.001
0150213001.005.001
0150214001.005.001
0150215001.005.001
0150216001.005.001
0150217001.005.001
0150218001.005.001
0150219001.005.001
0150220001.005.000
0150221001.005.001
0150222001.005.001
0150223001.005.000
0150224001.005.001
0150225001.005.001
0150226001.005.001
0150241001.005.001
0150242001.005.000
0150243001.005.001
0150244001.005.000
0150245001.005.001
0150246001.005.001
0150247001.005.001
0150248001.005.001
0150249001.005.001
0150250001.005.001
0150251001.005.001
0150252001.005.001
0150253001.005.001
0150254001.005.001
0150255001.005.001
0150256001.005.001
0150257001.005.001
0150258001.005.001
0150259001.005.000
0150260001.005.001
0150261001.005.001
0150262001.005.001
0150263001.005.001
0150264001.005.001
0150265001.005.001
0150266001.005.001
0150267001.005.001
0150268001.005.001
0150269001.005.001
0150270001.005.001
0150271001.005.001
0150272001.005.001
0150273001.005.001
0150274001.005.001
0150275001.005.001
01503000
01504000
01504500010020003
01505000010020003
01506000010020003
0150610001.005.000
01506200010040006
01506300010040006
0150632001.005.000
0150635001.005.000

```

DECISION
 ON
 SYSTEM
 BYTES

```

SIZEOFLSTIO := FCODEBYTES MOD EIGHTK;
% SHOW(" SIZEOFLSTIO = " C SIZEOFLSTIO FOR * DIGITS,TRUE);
FIRSTTIME := TRUE;
OFFSET := 0;
I := 0;
J := 0;
DO
  BEGIN
    IF J <= 47 THEN
      BEGIN
        RSLT := USERMAINTREQUEST(CTLUNIT,DOWNLOADMODE7,8192,OFFSET,0,
          IMLBUF2[I,*],MRD,HDPRESULT);
        % SHOW("1I = " C I FOR * DIGITS,TRUE);
        % SHOW("1OFFSET = " C OFFSET FOR * DIGITS,TRUE);
        IF RSLT > 0 THEN
          BEGIN
            SHOWRSLT(RSLT,LOADSLAVEIMLV);
            SHOWMRDBITS;
            SHOWHDPRESULT;

            SHOW("<< Microcode(1) NOT loaded!! >>",TRUE);
            IF NOT RELEASETARGET (OPTODO) THEN
              GO GRANDXIT;
          END;
        END
      ELSE
        BEGIN
          IF FIRSTTIME THEN
            I := 0;
            FIRSTTIME := FALSE;
            RSLT := USERMAINTREQUEST(CTLUNIT,DOWNLOADMODE7,8192,OFFSET,0,
              IMLBUF3[I,*],MRD,HDPRESULT);
            % SHOW("2I = " C I FOR * DIGITS,TRUE);
            % SHOW("2OFFSET = " C OFFSET FOR * DIGITS,TRUE);
            % SHOW("2J = " C J FOR * DIGITS,TRUE);
            IF RSLT > 0 THEN
              BEGIN
                SHOWRSLT(RSLT,LOADSLAVEIMLV);
                SHOWMRDBITS;
                SHOWHDPRESULT;

                SHOW("<< Microcode(2) NOT loaded!! >>",TRUE);
                IF NOT RELEASETARGET (OPTODO) THEN
                  GO GRANDXIT;
                END;
              END;
            NUMBROFIOS := * - 1;
            OFFSET := * + EIGHTK;
            I := * + 1;
            J := * + 1;
          END
        UNTIL NUMBROFIOS = 0;
        % SHOW(" NUMBROFIOS = " C NUMBROFIOS FOR * DIGITS,TRUE);
        % SHOW(" I = " C I FOR * DIGITS,TRUE);
        % SHOW(" J = " C J FOR * DIGITS,TRUE);
        % SHOW(" OFFSET = " C OFFSET FOR * DIGITS,TRUE);
        IF J < 47 THEN
          BEGIN
            RSLT := USERMAINTREQUEST(CTLUNIT,DOWNLOADMODE7,8192,OFFSET,0,
              IMLBUF2[I,*],MRD,HDPRESULT);
            % SHOW("3I = " C I FOR * DIGITS,TRUE);
            % SHOW("3OFFSET = " C OFFSET FOR * DIGITS,TRUE);
            IF RSLT > 0 THEN
              BEGIN
                SHOWRSLT(RSLT,LOADSLAVEIMLV);
                SHOWMRDBITS;
                SHOWHDPRESULT;

                SHOW("<< Microcode(3) NOT loaded!! >>",TRUE);
                IF NOT RELEASETARGET (OPTODO) THEN
                  GO GRANDXIT;
                END;
              END;
            END
          END
        END
      END
    END
  END

```

```

01506400010040006
0150640201.005.000
0150640401.005.000
01506410010040006
01506415010040006
0150641601.005.000
01506420010040006
01506430010040006
0150643501.005.000
0150643601.005.000
01506440010040006
0150645001.005.000
0150645201.005.000
0150645401.005.000
0150645601.005.000
0150645801.005.000
0150646001.005.000
0150646201.005.000
0150646401.005.000
0150646601.005.000
0150646801.005.000
0150647001.005.000
0150647201.005.000
0150647401.005.000
0150647601.005.000
0150647801.005.000
0150648001.005.000
0150648201.005.000
0150648401.005.000
0150648601.005.000
0150648801.005.000
0150649001.005.000
0150649201.005.000
0150649401.005.000
0150649601.005.000
0150649801.002.006
0150650001.002.006
0150650201.002.006
0150650401.002.006
0150650601.002.006
0150650801.002.006
0150651001.005.000
0150651201.002.006
0150651401.002.006
0150651601.002.006
0150651801.005.000
0150652001.002.006
0150652201.002.006
0150652401.002.006
0150652601.005.000
0150652801.002.006
0150653001.005.000
0150653201.005.000
0150653401.005.000
0150653601.005.000
0150653801.005.000
0150654001.005.000
0150654201.005.000
0150654401.005.000
0150655001.005.000
0150655201.005.000
0150655401.005.000
0150655601.005.000
0150655801.005.000
0150656001.005.000
0150656201.005.000
0150656401.005.000
0150656601.005.000
0150656801.005.000
0150657001.005.000
0150657201.005.000

```

Choose 1st Two-Dimens
Array
393 216 Bytes

Choose 2nd Two
Dimens. Array
393 216 Bytes

```

        END;
    END
ELSE
    BEGIN
        RSLT := USERMAINTREQUEST(CTLUNIT,DOWNLOADMODE7,4224,OFFSET,0,
            IMLBUF3[I,*],MRD,HDPRESULT);
%       SHOW("4I = " C I FOR * DIGITS,TRUE);
%       SHOW("4OFFSET = " C OFFSET FOR * DIGITS,TRUE);
%       SHOW("4J = " C J FOR * DIGITS,TRUE);
        IF RSLT > 0 THEN
            BEGIN
                SHOWRSLT(RSLT,LOADSLAVEIMLV);
                SHOWMRDBITS;
                SHOWHDPRESULT;

                SHOW("<< Microcode(4) NOT loaded!! >>",TRUE);
                IF NOT RELEASETARGET (OPTODO) THEN
                    GO GRANDXIT;
            END;
        END;
    END
ELSE
    RSLT := USERMAINTREQUEST(CTLUNIT,LOADSLAVEIMLV,FCODEBYTES,0,0,
        IMLBUF,MRD,HDPRESULT);
%       SHOW("5I = " C I FOR * DIGITS,TRUE);
%       SHOW("5OFFSET = " C OFFSET FOR * DIGITS,TRUE);
    IF RSLT > 0 THEN
        BEGIN
            SHOWRSLT(RSLT,LOADSLAVEIMLV);
            SHOWMRDBITS;
            SHOWHDPRESULT;

            SHOW("<< Microcode(5) NOT loaded!! >>",TRUE);
            IF NOT RELEASETARGET (OPTODO) THEN
                GO GRANDXIT
            ELSE
                GO NEXTDRIVE;
        END;
    END;

    IF EXPRESSMODE THEN
        SHOW("Download complete. Waiting 20 seconds for prom burn to " C
            CTLUNIT FOR * DIGITS,TRUE)
    ELSE
        SHOW("Download complete. Waiting 60 seconds for prom burn to " C
            CTLUNIT FOR * DIGITS,TRUE);
        SHOW("Do not power off or alter drive " C
            CTLUNIT FOR * DIGITS,TRUE);

% The SCSI disk drive has now turned off its SCSI interface.
% It won't come alive until after it has done the power up
% confidence tests (approx. 60 seconds). The second ATTRIBUTES command
% can then be issued to display the new firmware level. NOTE: If the
% second ATTRIBUTES command is issued before the target sequences
% (powers itself back up) the target will hang and the program will show
% an error.

    IF NOT EXPRESSMODE THEN
        SHOW("00:10 - Waiting for prom burn to target " C
            CTLUNIT FOR * DIGITS,TRUE);
        TIMER := 0;

    IF NOT EXPRESSMODE THEN
        DO BEGIN
            WHEN(10);
            N := (TIMER:=+1)*10 + 10;           % Seconds
            IF NOT EXPRESSMODE THEN
                SHOWRSLT(RSLT,-N);
            END UNTIL N >= 1*60;                % Drop dead time

    IF EXPRESSMODE THEN                        % WAIT 20 SECONDS ONLY
        WHEN (20);

```

```

0150657401.005.000
0150657601.005.000
0150657801.005.000
0150658001.005.000
0150658201.005.000
0150658401.005.000
0150658601.005.000
0150658801.005.000
0150659001.005.000
0150660001.005.000
0150662001.005.000
0150664001.005.000
0150666001.005.000
0150668001.005.000
0150670001.005.000
0150672001.005.000
0150674001.005.000
0150676001.005.000
0150678001.005.000
0150680001.005.000
0150682001.002.006
0150684001.002.006
01507000
01508000010040006
0150820001.005.000
0150840001.005.000
01509000
01510000
01511000
01512000
01513000
01514000
0151500001.005.000
01516000
01517000
01518000
01519000
01520000
01521000
01521200010020003
01521400010020003
01521600010020003
01521800010020003
01522000010020003
01523000010020003
01524000
01525000
01526000
01527000
01528000
01529000
01530000
01531000
01532000
01533000
01534000
01534500010020003
01535000010020003
01536000010020003
01537000
01538000
01538500010020003
01539000
01540000
01541000
01541500010020003
01542000010020003
01543000
01543200010020003
01543400010020003
01543600010020003

```

LOAD SINGLE
ARRAY

REPLACE SHORTBUF [0] BY NUL FOR SIZE (SHORTBUF);	01543620010040006
RSLT := USERMAINTREQUEST(CTLUNIT,TESTUNITREADY,0,0,0,	01543640010040006
SHORTBUF,MRD,HDPRESULT);	01543660010040006
IF RSLT > 0 THEN	01543680010040006
BEGIN	01543700010040006
SHOWRSLT(RSLT,LOADSLAVEIMLV);	01543720010040006
SHOWMRDBITS;	01543740010040006
SHOWHDPRESULT;	01543760010040006
SHOW("<< Test Unit Ready Failed!! >>",TRUE);	01543780010040006
IF NOT RELEASETARGET (OPTODO) THEN	01543800010040006
GO GRANDXIT;	01543820010040006
END;	01543840010040006
DISPLAY(" Test Unit Ready - OK ");	01543860010040006
REPLACE SHORTBUF [0] BY NUL FOR SIZE (SHORTBUF);	01543880010040006
RSLT := USERMAINTREQUEST(CTLUNIT,ATTRIBUTESV,254,0,0,	01543900010040006
SHORTBUF,MRD,HDPRESULT);	01544000010040006
IF RSLT > 0 THEN	01545000
BEGIN	01546000
SHOW("MCP interface error " C	01547000
RSLT FOR * DIGITS C	01548000
" after FW download" C	01549000
" timeout on drive " C	01550000
CTLUNIT FOR * DIGITS,TRUE);	01551000010020003
SHOWRSLT(RSLT,ATTRIBUTESV);	01552000
SHOW("Check the drive for malfunction",TRUE);	01553000
IF EXPRESSMODE THEN	01554000
SHOW(">> Run DFAST again NOT using Express Mode <<",TRUE);	01555000
GO GRANDXIT;	01556000
END;	01557000
SHOW("Target " C CTLUNIT FOR * DIGITS C " completed download",TRUE);	01558000
SHOWATTRIBUTES;	01558200010020003
% Get attributes of Target	01558400010020003
RSLT := USERMAINTREQUEST(CTLUNIT,INQUIRY,23,192,0,	01559000
SHORTBUF,MRD,HDPRESULT);	01560000010020003
IF RSLT > 0 THEN	01561000
BEGIN	01563000010020003
SHOW("MCP interface error " C	01565000
RSLT FOR * DIGITS C	01566000
" to get target" C	01571000
" attributes for SCSI drive " C	0157100201.005.000
CTLUNIT FOR * DIGITS,TRUE);	0157100401.005.000
SHOWRSLT(RSLT,ATTRIBUTESV);	0157100601.005.000
SHOW("Check TARGET for problem",TRUE);	0157100801.005.000
IF NOT RELEASETARGET (OPTODO) THEN	0157101001.005.000
GO GRANDXIT	0157101201.005.000
ELSE	0157101401.005.000
GO NEXTDRIVE;	0157101601.005.000
END;	0157101801.005.000
SHOWINQUIRYDATA;	0157102001.005.000
IF EXPRESSMODE THEN	0157102201.005.000
SHOW("ProdID-" C TPRODID FOR DRIVEIDLNG C "oldFWlv1-" C	0157102401.005.000
OLDFWLEVEL [0] FOR NEWFWLEVELNG C ".NewFWlv1-" C	0157102601.005.000
TPRODREVLVL FOR NEWFWLEVELNG,TRUE);	0157102801.005.000
% Release the target drive	0157103001.005.000
IF NOT RELEASETARGET (OPTODO) THEN	0157103201.005.000
GO GRANDXIT;	0157103401.005.000
	0157103601.005.000
	0157103801.005.000
	0157104001.005.000
	0157104201.005.000
	01571100010020003
	01571200010020003
	01571300010020003
	01571400010020003
	01571500010020003
	01572000
	01573000
	01574000
	01575000
	01575200010020002

SHOW("Okay to UR- target " C	01575400010020002
CTLUNIT FOR * DIGITS C	01575600010020002
" and select another drive.",TRUE);	01575800010020002
NEXTDRIVE:	01576000
END; % WHILE TRUE DO	01577000
END; % Load SCSI drive Firmware	01578000
\$\$ PAGE	01579000
	01580000
	01581000
	01582000
*****	01583000
% Initialization Block	01584000
*****	01585000
PROCEDURE INITIALIZE;	01586000
BEGIN	01587000
	01588000
INTEGER N;	01589000
	01590000
ODTMODE := MYSELF.INITIATOR >= 0;	01591000
ATODT := MYSELF.INITIATOR = 0;	01592000
	01593000
SDMASSIGNED :=	01594000
CTLASSIGNED := FALSE;	01595000
	01596000
TSTMP[0] := COMPILETIME(15); % mmddyy	01597000
TSTMP[1] := COMPILETIME(09); % hhmmss	01598000
TSTMP[2] := TIME(15); % mmddyy	01599000
TSTMP[3] := TIME(09); % hhmmss	01600000
	01601000
% See if we are privileged and can make use of the MCP interface.	01602000
	01603000
N := LINKLIBRARY(MCP);	01604000
IF DEBUG THEN	01605000
BEGIN	01606000
START "MCP linkage result = ",	01607000
N FOR * NUMERIC;	01608000
IF N = 1 THEN	01609000
ADD " (normal)";	01610000
SHOWIT(TRUE);	01611000
END;	01612000
IF N < 0 THEN	01613000
BEGIN	01614000
SHOW("FW download not possible. Program is ENDING.",TRUE);	01615000
GO GRANDXIT;	01616000010010001
END;	01617000
	01618000
BLANK;	01619000
START	01620000
MYSELF.NAME,	01621000
" (Version ",	01622000
COMPILETIME(20) FOR 2 DIGITS,	01623000
".",	01624000
COMPILETIME(21) FOR 3 DIGITS,	01625000
") compiled ",	01626000
ETSTMP[00] WITH DATETIME,	01627000
", run on ",	01628000
ETSTMP[12] WITH DATETIME;	01629000
SHOWIT(TRUE);	01630000
SHOWIT(FALSE); % Blank line	01631000
	01632000
START	01633000
"DO YOU WANT TO VIEW THE USER DOCUMENTATION FOR DFEST? 'Y' OR 'N'";	01634000
PROMPT;	01635000
IF PL = "Y" THEN	01636000
USERPROCESS	01637000
ELSE	01638000010020002
READFCN;	01638200010020002
	01638400010020002
	01639000
START	01640000
" >> Class C Material. Proprietary to Unisys Corporation <<";	01641000

```

SHOWIT(TRUE);
SHOWIT(FALSE);

$$ SET OMIT

START
" >>***** A T T E N T I O N *****<< ";
SHOWIT(TRUE); SHOWIT(FALSE);
%*IF NOT ATODT THEN % AT A TERMINAL
%* BEGIN
START
" >> This is an U N O F F I C I A L version of the FW << ";
SHOWIT(TRUE); SHOWIT(FALSE);
START
" >> download program to be used for project development << ";
SHOWIT(TRUE); SHOWIT(FALSE);
START
" >> ONLY. Do not pass this file to other organizations. <<";
SHOWIT(TRUE); SHOWIT(FALSE);
%* END
% $$ SET OMIT
ELSE % AT AN A-SERIES CONSOLE
BEGIN
START
" >> ONLY. Do not pass this file to other organizations. <<";
SHOWIT(TRUE); SHOWIT(FALSE);
START
" >> download program to be used for project development << ";
SHOWIT(TRUE); SHOWIT(FALSE);
START
" >> This is an U N O F F I C I A L version of the FW << ";
SHOWIT(TRUE); SHOWIT(FALSE);
END;
% $$ POP OMIT

START
" >>***** A T T E N T I O N *****<< ";
SHOWIT(TRUE); SHOWIT(FALSE);

$$ POP OMIT

END; % PROCEDURE INITIALIZE

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Select the Option Desired %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

PROCEDURE SELECTOPTION;
BEGIN
% Determine our operating mode

FUNCTION := 0;
EXPRESSMODE := FALSE;
DEBUG := FALSE;
SDMASSIGNED := FALSE;
CTLASSIGNED := FALSE;

WHILE FUNCTION = 0 DO
BEGIN
START
"Enter function [Ctlload,Devload,Verifyfile,Quit]";
PROMPT;
IF LL > 0 THEN
BEGIN
IF PL = "Q" THEN GO GRANDXIT;
IF PL = "DB" THEN % Debug Mode - extended displays
BEGIN
DEBUG := TRUE;
PL := * + 2;
LL := * - 2;

```

```

01642000
01643000
01644000
01645000
01646000
01647000
01648000
01649000
01650000
01651000
01652000
01653000
01654000
01655000
01656000
01657000
01658000
01659000
01660000
01661000
01662000
01663000
01664000
01665000
01666000
01667000
01668000
01669000
01670000
01671000
01672000
01673000
01674000
01675000
01676000
01677000
01678000
01679000
01680000
01681000
01682000
01683000
01684000
01685000
01686000
01687000
01688000
01689000
01690000
01691000
01692000
01693000
01694000010020003
01694200010020003
01694400010020003
01694600010020004
01694800010020004
01695000
01696000
01697000
01698000
0169900001.005.000
01700000
01701000
01702000
01703000
01704000010020003
01705000
01706000
01707000
01708000

```

```

END
ELSE IF PL = "XM" THEN      % Express Mode - Limit displays
  BEGIN
    EXPRESSMODE := TRUE;
    PL := * + 2;
    LL := * - 2;
    END;
  IF PL = "C" THEN FUNCTION := LOADCTLV      ELSE
  IF PL = "D" THEN FUNCTION := LOADEVV      ELSE
  IF PL = "V" THEN FUNCTION := VERIFYV      ELSE
    SHOW("Unrecognized function selector",TRUE);
  END; % LL > 0
END; % WHILE

END; % SELECTOPTION
$$ PAGE

*****
%      Epilog Procedure
*****
EPILOG PROCEDURE CLEANUP;
%-----
% Purpose:
%   Clean up any hanging threads.
%
% Parameters:
%   Epilog procedures cannot have parameters.
%
% Globals:
%   CTLATTACHED True: We have to un-attach it.
%-----
BEGIN
  REAL RSLT;

  IF CTlassigned THEN
    BEGIN
      SHOW("Epilog procedure invoked to detach from CTL " C
        CTLUNIT FOR * DIGITS,TRUE);
      RSLT := USERMAINTREQUEST(CTLUNIT,UNASSIGNCTLV,0,0,0,
        SHORTBUF,MRD,HDPRESULT);
      IF RSLT > 0 THEN
        BEGIN
          SHOW("Unable to detach from CTL " C
            CTLUNIT FOR * DIGITS      C
            ". Error "                  C
            RSLT FOR * DIGITS,TRUE);
          END;
        END; % Control attached

      IF SDMAssigned THEN
        BEGIN
          SHOW("Epilog procedure invoked to detach from drive " C
            CTLUNIT FOR * DIGITS,TRUE);
          RSLT := USERMAINTREQUEST(CTLUNIT,UNASSIGNUNITV,0,0,0,
            SHORTBUF,MRD,HDPRESULT);
          IF RSLT > 0 THEN
            BEGIN
              SHOW("Unable to detach from drive " C
                CTLUNIT FOR * DIGITS      C
                ". Error "                  C
                RSLT FOR * DIGITS,TRUE);
              END;
            END; % Disk attached
          END; % Epilog procedure
        $$ PAGE

*****
%      Outer Block
*****

```

```

01709000010020003
01710000010020003
01710100010020003
01710200010020003
01710300010020003
01710400010020003
01710500010020003
01711000
0171200001.005.000
01713000
01714000
01715000
01716000
01717000
01718000
01719000
01721000
01722000
01723000
01724000
01725000
01726000
01727000
01728000
01729000
01730000
01731000
01732000
01733000
01734000
01735000
01736000
01737000
01738000
01739000
01740000
01741000
01742000
01743000
01744000
01745000
01746000
01747000
01748000
01749000
01750000
01751000
01752000
01753000
01754000
01755000
01756000
01757000
01758000
01759000
01760000
01761000
01762000
01763000
01764000
01765000
01766000
01767000
01768000
01769000
01770000
01771000
01772000
01773000
01774000
01775000

```

INITIALIZE;	01776000
MAINLOOP:	01777000
WHILE TRUE DO	01778000
BEGIN	01779000
SELECTOPTION;	01780000
CASE FUNCTION OF	01781000
BEGIN	01782000
LOADCTLV:	01783000
OPTODO := UNASSIGNCTLV; % CTL PARAMETER FOR UMR	01784000
VERIFYFILE;	01785000
LOADCTLEW;	01785500010020004
LOADDEVV:	01786000
OPTODO := UNASSIGNUNITV; % DRIVE PARAMETER FOR UMR	01787000
VERIFYFILE;	0178800001.005.000
LOADDEVFW;	01788500010020004
VERIFYV:	01789000
VERIFYONLY := TRUE;	0179000001.005.001
VERIFYFILE;	01791000
VERIFYONLY := FALSE;	01791500010020004
END; % CASE FUNCTION OF	01792000
END; % WHILE TRUE	01792500010020004
GRANDXIT:	01793000
SHOW ("<< DFAST Program is ENDING >>", TRUE);	01794000010020004
IF CODE.OPEN THEN	01795000
CLOSE(CODE);	01796000
IF NOT ODTMODE THEN	01797000
CLOSE(RMT);	01798000010010001
CLOSE(LINE);	01799000
END.	01800000
	01801000
	01802000
	01803000
	01804000
	01805000